Trabajo Fin de Grado
Grado en Ingeniería de las Tecnologías de Telecomunicación. Intensificación en Electrónica

Desarrollo de un trazador de requisitos minimalista para diseños HDL

Autor: Carlos López Melendo

Tutor: Hipólito Guzmán Miranda

Dpto. Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2016

Trabajo Fin de Grado

Grado en Ingeniería de las Tecnologías de Telecomunicación.
Intensificación en Electrónica

Desarrollo de un trazador de requisitos minimalista para diseños HDL

Autor:

Carlos López Melendo

Tutor:

Hipólito Guzmán Miranda

Profesor Contratado Doctor

Dpto. Ingeniería Electrónica

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2016

Trabajo Fin de Grado: Desarrollo de un trazador de requisitos minimalista para diseños HDL

Autor: Carlos López Melendo
Tutor: Hipólito Guzmán Miranda

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Agradecer a mi familia y a Esperanza por su incansable apoyo durante toda esta etapa y a D. Hipólito Guzmán Miranda, tutor de este proyecto por darme a conocer este mundo de la Ingeniería de Requisitos así como enseñarme buenas prácticas en el desarrollo de proyectos.

Carlos López Melendo

Sevilla, 2016

Resumen

El proyecto realizado se encuadra dentro del marco de la Ingeniería de Requisitos así como de la Gestión de Requisitos y consiste en la implementación de un Trazador de Requisitos para diseños HDL. El objetivo de dicha aplicación es, una vez integrada con CMake y CTest, conocer el estado de nuestro proyecto en función de los requisitos y tests impuestos. La aplicación está compuesta por 3 partes, una primera basada en una interfaz web en la que se puede generar un fichero con los requisitos de nuestro proyecto; una segunda parte, basada en un script encargado de generar un informe con el estado de nuestro proyecto en función de los requisitos añadidos, los ficheros fuente y los ficheros de salida de CTest; por último, una tercera parte consistente en otra interfaz web en la que se muestra el estado de nuestro proyecto con todos los detalles del mismo.

Abstract

The present project belongs to the Requirements Engineering and Requirements Management disciplines. It consists in the implementation of a Minimalist Requirements Tracer for HDL Designs. The main objetive of the current application is, after being integrated with CMake and CTest, to obtain and represent the knowledge about the estate of a project considering its requirements and tests imposed. This application could be divided in three components. The first consists in an web interface where the user can write the requirements of the project and, after this, generate a csv-file whith them. The second part of this project is a script whose function is to generate a report with the state of the project, based on the requirements previously given and the tests which are in the project. Finally, the last part of the project is a web application. This application generates a web page where the user can know the state of the project.

Índice Abreviado

Índice

Notación

Acrónimos

TDD	Test Driven Design/Development
OSMRT	Open Source Requirements Management Tool
REM	Requirements Management
EDA	Electronic Design Automatization
FDA	Food and Drug Administration
FAA	Federal Aviation Administration
CMMI	Capability Maturity Model Integration
ISO	International Standars Organization
IEC	International Electrotechnical Commission
SPICE	Simulation Program with Integrated Circuits Emphasis
FPGA	Field Programmable Gate Array
HDL	Hardware Description Language
MINITRACER_PASS	Test de Minitracer que por defecto su salida es verdadero
MINITRACER_FAIL	Test de Minitracer que por defecto su salida es falso
MINITRACER_MANUAL	Test de Minitracer que por defecto su salida es falso y que el usuario debe validar externamente

1 Introducción

Todo lo que se hace se puede medir, sólo si se mide se puede controlar, sólo si se controla se puede dirigir y sólo si se dirige se puede mejorar.

Dr. Pedro Mendoza A.

En el siguiente capítulo se realiza una introducción al proyecto explicando de donde y cómo surge la idea de realizar un trazador de requisitos para diseños HDL y además se hace una introducción definiendo algunos términos que se han considerado relevantes y sobre los que hubo que buscar información antes de embarcarse en el proyecto. Dichos términos no son necesarios para entender el proyecto en sí o su desarrollo, pero sí que sirven para poder ver en qué rama de la ingeniería nos movemos y qué es lo que se denomina gestión de proyectos (ó "project management") y gestión de requisitos (ó "requirements management.")

1.1 La Idea

La idea de dicho proyecto surge por parte del profesor Hipólito Guzmán Miranda el cual, junto con el grupo de investigación del departamento de Ingeniería Electrónica del que forma parte, estiman conveniente el desarrollo de algún tipo de aplicación que, integrada con las ya existentes CMake y CTest que utilizan para verificación, les permita, de una manera sencilla, poder llevar un control del estado en el que se encuentran los proyectos que están realizando en base a los requisitos de los mismos. Por este motivo se inicia este proyecto asignado como trabajo fin de grado en el cual se investigan las opciones que existen a día de hoy en el mercado para dar solución al problema planteado y, una vez planteados los resultados, se decide llevar a cabo la implementación de la citada aplicación.

En primer lugar, el proyecto que se va a realizar, se puede encuadrar dentro de un doble marco, por un lado, el de la *Ingeniería de Requisitos*, y por otro el de la *Gestión de Proyectos*. Pero, ¿qué engloba tanto la *Ingeniería de Requisitos* como la *Gestión de Proyectos*? En las siguientes secciones se realizará una introducción a algunos conceptos relacionados con estos ámbitos.

1.2 Qué es la Ingeniería de Requisitos

Una definición de *"Ingeniería de Requisitos"* la podemos encontrar en el artículo *REASEM: Requirement management tool* y dice lo siguiente:

"La Ingeniería de Requisitos comprende las etapas fundamentales en el desarrollo de los productos de la ingeniería de software/firmware/hardware. Se centra en descubrir, analizar, escribir y verificar los servicios

y restricciones del sistema. Su importancia radica en que, de la definición de los requisitos depende el éxito de las etapas posteriores del desarrollo. Si los requisitos no se descubren o son encontrados en una etapa avanzada del desarrollo, esto provocará atrasos en el cronograma, aumento en el presupuesto, y el riesgo de que el producto no satisfaga las necesidades del cliente. De ahí la necesidad de proponer metodologías que permitan la captura y tratamiento de requisitos de una manera sistemática, oportuna y confiable soportadas por herramientas adecuadas."[17]

1.3 Qué es la Gestión de Proyectos

La Gestión de Proyectos es una disciplina que abarca la organización, el planeamiento, la motivación y el control de los recursos con la finalidad de alcanzar los objetivos propuestos para lograr el éxito en uno o varios proyectos dentro de las limitaciones establecidas. Pero más allá de todo esto, esta disciplina abarca algunos términos más que interesa conocer ya que, términos cono Gestión de Requisitos, Trazabilidad, Trazabilidad de Requisitos o Matriz de Trazabilidad son de interés para el desarrollo de este proyecto.

1.3.1 Gestión de Requisitos [2]

Es el proceso de documentación, análisis, trazado, priorización y acuerdo de requisitos junto con su control de cambios e información a las principales partes interesadas. Es continuo a lo largo de la vida de un proyecto.

El objetivo de la Gestión de Requisitos es asegurarse de que se realiza una organización de documentos, se verifica y se conocen las necesidades y expectativas, tanto de los consumidores como de los accionistas, internos o externos.

El proceso de se inicia con el análisis y la obtención de los objetivos y las limitaciones existentes pero no se queda solo ahí, va más allá, integra a los requisitos y a la organización para trabajar conjuntamente y poder ir adaptando estos en función de las necesidades, ya que estas pueden ir cambiando a lo largo de la vida del proyecto.

La Gestión de Requisitos involucra comunicación entre el equipo de proyecto y las partes interesadas y, además, el ajuste a los cambios en los requisitos en el transcurso del proyecto.

1.3.2 Trazabilidad [3]

El término Trazabilidad, del inglés "Traceability" se puede definir como *"ability to chronologically interrelate the uniquely identifiable entities in a way that matters"*. Esto quiere decir que cada requisito de nuestro proyecto tiene un origen (es causado por algo), que bien puede ser, una condición de un inversor, una limitación, o cualquier otro requisito que dé como resultado este otro. Esto es así porque en la Gestión de Requisitos se sigue una evolución estructural, es decir, el camino que sigue un requisito desde el "de dónde viene", pasando por el "cómo es testado" al "qué impacto tiene sobre el resultado".

"Trazabilidad de Requisitos", en términos generales, se considera una sub-disciplina de lo que sería la "Gestión de Requisitos" como lo pueden ser "software development" o "systems engineering". Está relacionada con la documentación de la vida de un requerimiento y provee de una trazabilidad bidireccional entre varios requisitos asociados. Permite al usuario encontrar el origen de cada requisito y poder "anotar" cada cambio que se produzca en el mismo. Por ello es necesario documentar cada cambio realizado en cada uno de los requisitos de nuestro proyecto.

Otra definición:

"Requirements traceability refers to the ability to describe and follow the life of a requirement, in both forwards and backwards direction (i.e. from its origins, through its development and specification, to its subsequent deployment and use, and through all periods of on-going refinement and iteration in any of these phases".[18]

1.3.3 Matriz de Trazabilidad [4]

Una matriz de trazabilidad es un documento, generalmente una tabla, que se usa para determinar relaciones, correlando cualesquiera dos "baselined [1] documents" usando lo que se denomina "a many-to-many relationship" [2] [19].

1.4 Tipos de Requisitos

Una vez introducidos los términos "Gestión de Proyectos", "Gestión de Requisitos", "Trazabilidad" y "Matriz de Trazabilidad", conviene hablar de los diferentes grupos en los que se puede clasificar los requisitos pertenecientes a un proyecto.

1.4.1 Requisitos Funcionales [5]

Este tipo de requisitos son los que se usan para describir la funcionalidad del sistema y/o de sus componentes. Si nuestro sistema final lo definimos según el criterio de "black-box model"[20], los requisitos funcionales se corresponderían con las entradas y salidas del mismo. Los Requisitos Funcionales normalmente son de la forma: "system must do <requirement>".

1.4.2 Requisitos de Prestaciones [6]

También son conocidos como "Non-functional Requirements". Se pueden definir como las prestaciones que debe de tener el sistema, es decir, especifican aún más como deben de ser cada uno de los requisitos funcionales.

1.4.3 Requisitos de Diseño [7]

Este tipo de requisitos se definen como las limitaciones propias del diseño definidas por la propia naturaleza del mismo. Un ejemplo de este tipo de requisitos puede ser por ejemplo un requisito de dimensiones de un chip en función de la tecnología de diseño utilizada.

1.4.4 Design constraints [7]

De este tipo de requisitos no se puede hablar como tales en el propio sentido de la palabra, se definirían más bien como las limitaciones impuestas por el usuario. Estas limitaciones suelen ser tales como: tiempo de ejecución, presupuesto disponible, materiales que deben se usados, etc. No se pude decir que sean requisitos como se ha mencionado antes debido a su propia naturaleza y son tomados más bien como restricciones. Un ejemplo de restricción puede ser el coste máximo del proyecto.

1.5 TDD: Test-Driven-Design/Development [1]

Por último, se considera interesante introducir la metodología "Test-Driven-Design/Development" para el desarrollo de software y firmware ya que, aunque no tiene nada que ver con la Gestión de Requisitos, está ligado a estos ya que se considera un buen método para la escritura de código que sirva para cumplir uno o varios requisitos del proyecto y que además pueda pasar los tests asignados al mismo.

[1] A baseline is an agreed description of the attributes of a product, at a point in time, which serves as a basis for defining change.

[2] A many-to-many relationship refers to a relationship between tables in a database when a parent row in one table contains several child rows in the second table, and vice versa.

TDD es un proceso de desarrollo software en el que se repite un pequeño ciclo de desarrollo consistente en los siguientes pasos:

1. El desarrollador escribe un test: En este tipo de método, cada nueva aplicación comienza con la codificación de un test. Para escribir dicho test, el desarrollador debe tener claras las especificaciones y requisitos de la aplicación.

2. Ejecutar los tests y comprobar si fallan: Con esto se comprueba que el "test harness"[21] funciona correctamente, es decir, que el nuevo test programado, para el cual aún no se ha desarrollado ninguna función que dé solución al requisito planteado, no dé una salida correcta por error. Con ésto estamos probando el test en sí mismo ya que estamos comprobando que el test no genera una salida válida sin haberle pasado nada como entrada.

3. Escribir el código: El código escrito en este punto posiblemente pase el test, pero aun así, no será un código elegante. Esto no es un problema ya que, en el paso 5, dicho código será mejorado.

4. Probar si el código pasa el test: Si el nuevo código pasa el test en todos los casos, el programador puede estar seguro que de dicho código da solución al requisito a resolver. Si no los pasa, se debe re-programar dicho código.

5. "Refactor Code": En este paso se realiza la limpieza del código y se mejora el existente haciéndolo más eficiente. Dicho código se vuelve a probar y se comprueba que pasa los tests.

6. Se vuelve a repetir el ciclo codificando un nuevo test y volviendo a hacer todos los pasos.

2 Estado del Arte

Gestión es hacer las cosas bien, liderazgo es hacer las cosas.

PETER DRUCKER

En el siguiente capítulo se puede encontrar la información recabada después de la realización de un estudio del estado del arte referente al proyecto en el que nos hemos embarcado. En él, se realiza un recorrido por las principales herramientas que se han encontrado para la gestión de proyectos y requisitos clasificándolas en herramientas de pago y herramientas gratuitas.

2.1 Introducción al Estudio del Estado del Arte

En la mayoría de los casos, cuando se comienza con el diseño de un proyecto se incurre en el error de no tener claros o bien definidos y delimitados los requisitos del mismo, lo cual supone, por un lado, insatisfacción por parte del cliente, que no obtiene el resultado deseado, y por otro lado, y como consecuencia de lo anterior, un aumento en los costes de desarrollo del proyecto y el consiguiente retraso en las fechas de entrega. Por todo ello, se considera necesario el uso de algún tipo de herramienta con la que podamos tener presentes durante el desarrollo del proyecto todos los requisitos y el estado de los mismos así como la posibilidad de modificar, añadir o quitar requisitos y saber rápidamente el efecto que tienen esas modificaciones sobre nuestro proyecto.

Actualmente en el mercado podemos encontrar diversas herramientas software que nos permiten realizar un seguimiento del estado de nuestro proyecto.

Por un lado, existen encontrar aplicaciones software implementadas por empresas privadas como IBM, Mentor, Aldec, Polarion, MicroFocus, Hewlett Packard Enterprise o Visure entre otras con el inconveniente de que son herramientas de pago.

Por otro lado también hay algunas herramientas de código libre que se pueden obtener de manera gratuita. Podemos encontrar herramientas como pueden ser REASEM, OSRMT o REM, esta última presentada en la tesis doctoral “Un Entorno Metodológico de Ingeniería de Requisitos para Sistemas de Información” presentada por Amador Durán en septiembre de 2000.

2.2 Herramientas de código libre o gratuitas

2.2.1 CTest [8]

CTest es una herramienta de test implementada dentro de la herramienta CMake. CTest puede ser usado para automatizar la actualización, configuración o testeo y enviar los resultados a “Dart” o “CDash” que es un sistema de “dashboard” (este término se define más adelante) . Hay dos formas básicas de uso de CTest:

1. La primera[22] forma es usando CMake para configurar y compilar el proyecto, usando unos comandos especiales en el archivo “CMakeLists.txt” para crear los tests. CTest puede entonces ser usado para ejecutar los tests, y opcionalmente enviar sus resultados al “dashboard server”.

2. La segunda[23] forma es ejecutar un script con CTest para controlar tanto el proceso completo de testeo, configuración y compilación del proyecto como la propia ejecución de los test. En la siguiente sección se especifica qué es un “Dashboard”

2.2.1.1 Dashboards [9]

Los “Dashboards” son unos paneles en forma de página web en los que se pueden integrar información de tests software en diferentes máquinas/arquitecturas. Se pueden usar para ver si el equipo de proyecto avanza de una forma correcta en el testeo de las “user stories” (Agile) o de los requisitos (CMMI).

2.2.2 REASEM [10]

REASEM (Requisitos apoyados por la metodología ABC-Besoin-SEM) es una herramienta de soporte a la metodología ABC-Besoins-SEM, centrada en diseños de sistemas embebidos, que permite la interacción con el modelo de requisitos, conceptual y de diseño de dicha metodología. REASEM ofrece las funciones básicas de un software que apoya las fases de la Ingeniería de Requisitos tales como: permitir la identificación, clasificación y documentación de requisitos, y ayudar a la generación de un modelo conceptual y a su transformación en un modelo de diseño.

2.2.2.1 Arquitectura Básica de REASEM [10]

La arquitectura básica de la herramienta está conformada por tres capas:

1. Capa de comunicación: Contiene la interfaz gráfica, en la cual se estable la comunicación usuario-herramienta.

2. Capa de lógica de negocio: Se compone de todas las funciones pertenecientes a la metodología ABC-Besoins-SEM. Los tres módulos de esta capa son: el módulo del modelo de requisitos, donde sería posible ingresar requisitos pertenecientes a las diferentes categorías y sub-categorías de dichos modelos; el módulo del modelo conceptual que permite generarlo y relacionar sus componentes con los requisitos obtenidos y por último, dentro de la segunda capa, está el módulo del modelo de diseño el cual guiará al usuario en la construcción de un modelo de diseño especificado en términos básicos del lenguaje SystemC.

3. Capa de datos: almacena información referente a las fases de desarrollo de los sistemas embebidos, desde su definición, pasando por la definición de requisitos, la elaboración del modelo conceptual, hasta el modelo de diseño

2.2.3 OSRMT [11]

OSRMT (Open Source Requirements Management Tool) es una herramienta de software libre pensada para asistir en todo el ciclo de vida del desarrollo de un proyecto. Permite la descripción avanzada de diversos tipos de requisitos y garantiza la trazabilidad entre todos los documentos relacionados con la ingeniería de

requisitos (casos de uso, requisitos, casos de prueba). La última versión data de marzo del 2013 por lo que se desaconseja su uso debido a la posible falta de actualización de dicho software. La herramienta, además de una interfaz de escritorio junto a una interfaz web, integra módulos de administración y configuración, gestión de documentos de la Ingeniería de Requisitos, trazabilidad entre los documentos de trabajo e informes y estadísticas. Además de todo lo mencionado, el sistema también provee:

1. Gestión de la configuración: versionado y registro de los cambios realizados en los diferentes elementos.
2. Gestión de usuarios y permisos.
3. Herramientas de migración para los diversos cambios de versiones.
4. Múltiples idiomas (importación y exportación para dar soporte a diversos idiomas).
5. Importar y exportar información en XML y mediante línea de comandos.
6. Exportar información en HTML o mediante línea de comandos.
7. Informes:
    - Básicos.
    - Específicos creados por el usuario.
    - A partir de los resultados de búsquedas avanzadas.
    - Exportados a HTML PDF.

2.2.4 REM

Para introducir un poco lo que es “REM” hemos preferido hacer referencia directamente a la descripción que podemos encontrar en la página de descarga del departamento de lenguajes y sistemas informáticos de la Universidad de Sevilla:

“REM (Requirements Management) es una herramienta experimental gratuita de Gestión de Requisitos diseñada para soportar la fase de Ingeniería de Requisitos de un proyecto de desarrollo software de acuerdo con la metodología definida en la Tesis Doctoral "Un Entorno Metodológico de Ingeniería de Requisitos para Sistemas de Información", presentada por Amador Durán en septiembre de 2000”. [24] Como características principales podemos destacar:

1. Herramienta de gestión de recursos gratuita.
2. Interfaz de usuario sencilla y e intuitiva.
3. Basada en XML, XSLT y genera HTML.
4. Uso de plantillas y patrones lingüísticos para requisitos.
5. Almacenamiento en BB.DD relacional.
6. Se pueden tener varios proyectos abiertos y varias ventanas del mismo proyecto.

No obstante, la versión disponible para descarga se corresponde con la 1.2.2 que data del 25 de noviembre de 2004 por lo que es un proyecto cerrado y de uso no demasiado aconsejado debido a la falta de actualización con las necesidades y requisitos de hoy en día.

2.2.5 HELER

HELER (Herramienta Libre para la Especificación de Requisitos), ofrece soporte a actividades de Ingeniería de Requisitos contempladas en la fase de entendimiento del problema, enmarcada dentro del Proceso Unificado

[25]. A continuación se pasa a explicar la estructura de la herramienta por módulos: (Se cita directamente la definición de cada módulo dada en la sección de la revista [25] que habla sobre HELER).

La herramienta implementa una arquitectura que consta de los módulos que a continuación se describen:

1. Módulo proyecto: Aquí se crea el proyecto, se registran los objetivos del proyecto que se pretende desarrollar, se gestiona la visión identificando las necesidades y características de sistema y se define el glosario donde se explica el significado de cada uno de los términos provenientes de las áreas del proyecto.

2. Módulo stakeholder : Este módulo se encarga de gestionar lo relacionado con los stakeholders y participantes del proyecto, reúne las funcionalidades de crear, modificar, consultar y eliminar la información general del stakeholder, el rol de cada uno y sus actividades, asignación de fuentes e historial, en este módulo también se asigna la prioridad a los stakeholders, se determina el poder de decisión que tienen éstos para solicitar, aprobar, modificar y eliminar requisitos.

3. Módulo actor y caso de uso: Aquí se especifican y describen actores y casos de uso, se revelan las actividades que ejecuta cada uno de estos y la interacción con otros actores, por medio de un flujo de eventos, donde se describen por orden las acciones y las responsabilidades tanto de los actores como del sistema. Se documenta el alcance del caso de uso mediante pre-condiciones y post-condiciones a un nivel de abstracción de fácil compresión y empleando el lenguaje de negocio, se determinan los requisitos no funcionales de cada caso de uso, la prioridad, las excepciones, flujos alternos que pueden ocurrir durante la ejecución del flujo de eventos y se enumeran los documentos, fuentes y demás actividades que se emplearon para especificar cada caso de uso.

4. Módulo de requisito: Permite identificar y comprender los requisitos. Esta identificación incluye determinar el tipo de requisito, especificación e importancia que tiene un requisito en términos de implementación a través de la asignación de prioridad, urgencia y estado. Se valida si los requisitos creados satisfacen los objetivos mediante la matriz de trazabilidad que permite describir y seguir la vida de un requisito.

2.3 Herramientas de pago

2.3.1 IBM Rational DOORS [12]

La aplicación desarrollada por IBM para la gestión de requisitos ha sido llamada “IBM Rational DOORS”. El fin de dicha aplicación es optimizar la comunicación, colaboración y verificación de requisitos dentro de un proyecto. Dicha solución es escalable y ayuda a gestionar el ámbito y los costes del proyecto y alcanzar los objetivos empresariales. Permite capturar, rastrear, analizar y gestionar cambios en la información, así como demostrar la conformación con las normativas y estándares. Las características principales de dicha herramienta son:

1. Gestión de requisitos: proporciona un amplio entorno de gestión de requisitos.

2. Rastreabilidad: vincula los requisitos a los elementos de diseño, los planes de prueba, los casos de prueba y otros requisitos.

3. Escalabilidad: es escalable para afrontar las necesidades de gestión de requisitos en constante cambio.

4. “Test Tracking Toolkit” o kit de herramientas para el seguimiento de pruebas: incluye dicha herramienta para entornos de prueba manuales, de manera que quienes realicen las pruebas (“testers”) puedan vincular requisitos a casos de prueba.

5. Integraciones: gestiona los cambios en los requisitos con un sencillo sistema de propuesta de cambios predefinidos o bien con un flujo de trabajo de control de cambios personalizado más completo mediante la integración con las soluciones de gestión de cambios de Rational.

2.3.2 Aldec [13]

Aldec es una compañía líder en Automatización del Diseño Electrónico (EDA) que trabaja para innovar en soluciones para la creación de diseños, simulaciones y verificación que ayuden en el desarrollo de diseños FPGA, ASIC, SoC y sistemas embebidos complejos.

El producto que nos presenta dicha compañía recibe el nombre de "Spec-TRACER" y lo primero que se puede ver y que lo diferencia del resto de software presentado es que este sí es específico para diseños HDL. Haciendo una referencia directa al datasheet de dicho software pasamos a hacer un breve resumen de la aplicación:

"Spec-TRACER is a unified requirements lifecycle management application designed specifically for FPGA and ASIC designs. Spec-TRACER facilitates requirements capture, management, analysis, traceability and reporting that seamlessly integrates with HDL design and simulation tools. Traceability links between requirements and elements of the HDL design, testbench, log files and waveforms are established easily and upstream/downstream traceability reports are generated automatically. Spec-TRACER helps manage, control and track requirements from specification to requirements, concept to design and verification plans to results."[26]

En cuanto a las características de este software, en primera instancia podemos destacar que soporta la normativa "DO-254" para el desarrollo de aplicaciones firmware aeroespaciales. En cuanto al resto de características podemos destacar las siguientes:

1. Es posible la importación de requisitos desde documentos Excel, Word, o archivos generados con la aplicación DOORS de IBM.
2. Trazabilidad para diseños HDL y "testbench".
3. Gestión de tests.
4. Análisis de impacto de cambios sobre el proyecto.
5. Informes predefinidos y también definidos por el usuario.
6. Soporta ventanas de simulación HDL como pueden ser ModelSim o Active-HDL entre otros.

2.3.3 Mentor

Por su lado, la empresa Mentor Graphics tiene su propio software de Requirements Tracing que también se centra, entre otros, en proyectos HDL. En cuanto a la descripción que hacen de dicho producto, hablan de que el mismo realiza la captura de requisitos desde el sistema al componente y además incluye trazabilidad de requisitos lo cual asegura que existen relaciones entre los requisitos, el código que los implementa, los test y los resultados que demuestran la verificación de dichos requisitos.

En cuanto a las características principales que ofrece Mentor cabe destacar las siguientes:

1. Control y predicción de los tiempos del proyecto.
2. Trazabilidad de requisitos a través del proceso de diseño hardware.
3. Enlace de todos los documentos del proyecto y las herramientas de diseño HDL.
4. Predicción de consecuencias sobre el proyecto debidas a las modificaciones en las especificaciones y/o requisitos.
5. Soporta la norma DO-254.

2.3.4 Polarion Requirements [14]

Polarion Requirements es un software de "Requirements Management" creado por la empresa Polarion y que la misma define como la solución completa para la gestión de los requisitos de un proyecto durante todas las etapas/fases del mismo. Este software da soporte a numerosas metodologías y flujos de trabajo. Soporta las metodologías Agile/Lean, Tradicional, Híbrida o entornos personalizados. A la vez, como es personalizable, permite de manera sencilla usarlo en gran variedad de proyectos que sigan cualquier tipo de estándar industrial, incluyendo FDA, FAA, CMMI, ISO, IEC, SPICE, entre otros. Como opciones destacadas de Polarion Requirements podemos citar desde su datasheet las siguientes:

1. "Browser-based Solution: Proporciona acceso 24/7 al proyecto para equipos distribuidos en diferentes sitios.
2. Live Documents: Distribuyes información clave a los "stakeholders" en tiempo real.
3. Email Alerts: Mejora el intercambio de información y el control de riesgos.
4. Offline Collaboration: Soporta intercambio de información vía Polarion Round-trip™ para Microsoft® Word/Excel®
5. Approval Center: Rápida revisión con aprobación/desaprobación en 1-click además de un hilo de debate."[27]

2.3.5 Micro Focus

Micro Focus es una compañía que se define a sí misma como una empresa que ofrece soluciones de software innovadoras que permiten a las compañías desarrollar, probar, desplegar, evaluar y modernizar aplicaciones empresariales vitales para el negocio. En cuanto a su software de "Requirements Management", tienen 2 distribuciones en función de las necesidades. Dichas 2 distribuciones son:

1. Atlas: "Micro Focus Atlas is an Agile requirements and delivery platform that enables teams to create products in a much more collaborative and flexible way in comparison to other requirements management tools. Atlas blends the definition of business needs with iterative Agile delivery, without forcing process change on delivery teams. It provides the ability to track progress of remote development teams in the context of defined requirements to assure the right products for the business are progressing as needed. Atlas helps organizations of all sizes gather, define, plan, track and deliver complex products to market with speed and confidence." [28]
2. Caliber: "Micro Focus® Caliber helps address these challenges by ensuring that all stakeholders are effectively working towards the same goals by means of a shared view of requirements from inception through delivery. A unified approach to requirements visualization, definition and management ensures alignment of business objectives with development deliverables as project teams collaborate to prioritize, approve and understand the impacts of real-time change."[29]

2.3.6 Hewlett Packard [15]

La empresa Hewlett Packard ofrece un recurso que ellos llaman "Requirements Management" el cual ofrece a sus equipos distribuidos un repositorio único y compartido para colaborar y compartir requisitos, entender sus relaciones para las pruebas y evaluar los defectos conectados. Además, incluye requisitos múltiples preconfigurados. Dicho recurso es un componente integrado de: "HPE Quality Center" y "HPE Application Lifecycle Management"

2.3.6.1 HPE Quality Center y HPE Application Lifecycle Management

De los datasheet de ambas aplicaciones se han extraído las principales características de las mismas que a la vez, son comunes entre "HP Quality Center" y "HPE Application Lifecycle Management" con la diferencia

de que la primera está mas centrada en el control del desarrollo global del proyecto y la segunda, se centra en el ciclo de vida de los requisitos y tests:

1. “ Standarización de la definición de requisitos a través de plantillas configurables.
2. Potente editor de texto.
3. Habilidad para observar la cobertura de los requisitos en el proyecto y la versión en la que se encuentran.
4. Posibilidad de trazar relaciones entre requisitos y procesos, ficheros del proyecto, defectos y tests.
5. Relaccionar directamente requisitos y tests para facilitar el alineamiento con los cambios producidos en los mismos.
6. Control de versiones.”[30] [31]

2.3.7 Visure Requirements [16]

Visure es una compañía que se centra en la Ingeniería de Requisitos orientada al Proceso. Dicha compañía implementa un software al que le dan el nombre de “Visure Requirements” y el cual describen como una solución que integra el soporte a proceso, la colaboración y la calidad en una única plataforma centralizada promoviendo, a la vez, la reutilización.

Además de lo ya dicho, la empresa describe su producto como una ayuda para estandarizar e implementar el proceso de Definición y Gestión de Requisitos en toda la organización, estableciendo una estructura de especificación de requisitos común y permitiendo gestionar los cambios a lo largo de todo el ciclo de vida. En cuanto a la lista de funcionalidades[32] de “Visure Requirements” tenemos la siguiente:

1. Captura de requisitos: Captura manual y automática de requisitos de diferentes fuentes.
2. Análisis de requisitos: Creación del modelo de dominio para entender el negocio. Contextualización de los requisitos con el modelo de dominio del problema.
3. Análisis semántico: Realización de un análisis semántico de los requisitos.
4. Especificación de la solución: Modelado de las relaciones entre el sistema y las entidades externas.
5. Validación de la especificación: Trazabilidad completa entre los elementos.
6. Verificación y pruebas de la aceptación: Definición de las pruebas y los criterios de aceptación.
7. Gestión de requisitos: Gestión de atributos, flujos de trabajo, trazabilidad, filtros y vistas orientadas al usuario.
8. Trazabilidad de requisitos: Trazabilidad completa de principio a fin.
9. Gestión del proyecto: Organización del proyecto y del producto.
10. Generación de informes: Generación de informes avanzados con los datos del proyecto.
11. Soporte a reutilización: Apoyo a la reutilización sencilla de requisitos y pruebas asociadas, así como de procesos complejos, para familias de productos y variantes.
12. Gestión de la configuración: Gestión de la configuración para versiones de elementos individuales y especificaciones complejas.
13. Integraciones con otras herramientas: Integración con las herramientas más notorias de diseño, de pruebas, de gestión de proyecto y de gestión de usuarios, y por último, APIs y PluginSDK abiertos para personalizar.

2.4 Conclusión

Una vez realizado este estudio del estado del arte, se puede observar que existen multitud de herramientas para la gestión de requisitos y, en general, la gestión de proyectos pero, en el caso más concreto de proyectos HDL, este número se ve reducido drásticamente a tan sólo dos aplicaciones diseñadas en particular para este fin, *Spec-Tracer,* perteneciente a la compañía *Aldec* y *ReqTracer,* perteneciente a la compañía *Mentor,* y una tercera, *IBM Rational DOORS,* perteneciente a la compañía *IBM* que se podría adaptar para este tipo de diseños. Esto, unido a que son programas bajo licencia de pago, hacen razonable la idea de la realización de un trazador de requisitos minimalista para diseños HDL y por este motivo se decide emprender este proyecto, para la realización de un software de gestión de requisitos que sea útil para diseños HDL y que además, sea software libre y gratuito.

Hacer notar también que en el mercado existen multitud de herramientas para la gestión de proyectos que, aunque todas se reducen a unas características comunes, cada una a parte, tiene unas características propias que las diferencian del resto. Por ello, solo se han estudiado en profundidad y añadido a este estudio del estado del arte las que se han considerado más importantes siguiendo el criterio de su popularidad. No obstante, a continuación se nombran en una lista algunas del resto de herramientas más que se han conocido pero de las que no se ha realizado un estudio.

1. Borland CaliberRM.
2. Insoft Prosareq.
3. ViewSet PACE.
4. Igatech System RDT.
5. SpeeDEV RM.
6. RBC RMTrack.
7. Serena RTM.
8. Teledyne Brown XTie-Rt.
9. IRQA 4cubo.
10. RETO.
11. JEREMIA.
12. RAMBUTAN.

Como se menciona antes, de este estudio estado del arte también se pueden sacar una lista con características deseables que debe tener un trazador de requisitos basándose en las características comunes vistas en todas las aplicaciones que han sido estudiadas. A continuación se muestra dicha lista:

1. Exportar información en HTML.
2. Gestión de usuarios y permisos.
3. Importar y exportar información sobre los requisitos.
4. Posibilidad de tener varias ventanas abiertas con los resultados de diferentes proyectos.
5. Generación de informes.
6. Control de cambios.
7. Clasificación y visualización de requisitos.

8. Predicción de consecuencias debidas a los cambios en requisitos.
9. Generación de matriz de verificación.
10. Generación de matriz de trazabilidad.
11. Control y predicción de los tiempos del proyecto.
12. Integración con otras herramientas.
13. Generación de un resumen del estado del proyecto.
14. Identificación del número de tests pasados por cada requisito.
15. Identificación de posibles requisitos no recogidos en los documentos.
16. Identificación de tests inexistentes en el proyecto o en los informes.
17. Captura manual y automática de los requisitos de las diferentes fuentes.
18. Control de versiones.
19. Aplicación distribuida.

Como se puede comprobar, la citada lista incorpora elementos que incumplen con la condición de que el proyecto a realizar sea minimalista por lo que, no se van a implementar todas las características antes mencionadas, aún así se ha creído conveniente enumerarlas todas ya que han sido las cualidades más destacadas que se han extraído del estudio.

3 Requisitos de la aplicación

Una estructura organizativa pobre hace el buen trabajo imposible, no importa lo buenas que sean las personas

PETER DRUCKER

En el presente capítulo se estudiarán los requisitos necesarios para la realización del trazador de requisitos para diseños HDL partiendo de la base, que sería además el primer requisito, de que debe de ser una aplicación minimalista, es decir, que no se requieran de muchos recursos o aplicaciones externas en la máquina en la que se use dicho software, siendo éste portable a cualquier ordenador sin necesidad de realizar ningún tipo de operación para dicho fin, tan solo la copia de los ficheros en un medio extraíble.

A continuación se enumeran los requisitos agrupándolos en *requisitos necesarios* y *requisitos opcionales.*

3.1 Requisitos necesarios

1. Trazador de requisitos válido para diseños HDL.
2. Aplicación portable.
3. Software que no necesite de compilador.
4. Aplicación que sea capaz de leer los requisitos de un archivo de texto plano y/o de archivo *csv.*
5. Los requisitos pueden estar contenidos en uno o varios archivos de texto plano y/o archivos *csv.*
6. Aplicación integrable con CMake y CTest.
7. Aplicación capaz de leer los resultados de los test realizados por CTest.
8. Aplicación capaz de leer las etiquetas identificadoras de requisitos en los ficheros de código.
9. La aplicación deberá generar un archivo *csv* en el que presente todos los resultados del estado del proyecto.
10. La aplicación deberá gestionar de manera oportuna el anterior archivo *csv* de manera que presente una página html a modo de *dashboard* con toda la información extraída del estado del proyecto.
11. Dicha página html presentará un resumen del estado del proyecto a modo de barras de progreso.
12. La página html deberá dar información detallada del estado de todos los requisitos del sistema.

13. La página html presentará información detallada acerca de los test que se han pasado al proyecto.
14. Se deberá facilitar también una interfaz html para añadir requisitos al proyecto y que estos puedan ser exportados a un fichero *csv.*
15. La aplicación deberá generar una matriz de verificación que representará requisitos frente a test.
16. La aplicación generará una matriz en la que se presenten requisitos frene a ubicación de los mismos.
17. La página html generada deberá poder ser interpretada en local, utilizando un navegador web, sin necesidad de ningún servidor para la interpretación del contenido.
18. La aplicación deberá indicar si existen requisitos que no estén implementados.
19. La aplicación deberá indicar si existen tests que no estén implementados.
20. La aplicación deberá indicar si existen requisitos que han sido implementados pero no están recogidos en los archivos de texto plano y/o *csv.*

3.2 Requisitos opcionales

1. La interfaz html para añadir requisitos tendrá la opción de importar algún archivo *csv* con requisitos y añadir requisitos a los ya existentes.
2. La aplicación podrá reconocer requisitos no sólo en fichero *.vhd* sino también en archivos con cualquier otro tipo de extensión (siempre que sean de texto plano): *.c, .java, .py, .bash, .html, .js, .css, .v, .doc, etc.*
3. Uso de un código de colores en la página html para reconocer el estado. El código será:
    - Rojo para errores.
    - Naranja para warnings.
    - Verde si todo correcto.

4 Solución Propuesta

Una de las virtudes del ingeniero es la eficiencia.

Guang Tse

En el siguiente capitulo se procede a explicar en detalle la solución propuesta para la implementación del trazador de requisitos en función de los requisitos descritos en el capitulo anterior. Se explicará desde el flujo de trabajo seguido, pasando por las tecnologías empleadas para la realización del software junto con el motivo de por qué se han empleado ésas y no otras, además, se hace una descripción del proceso de realización del proyecto pasando ya por último a una descripción detallada de cada una de las partes de las que está compuesta el programa realizado.

4.1 Flujo de Trabajo

En esta sección de la memoria nos centramos en explicar el flujo de trabajo empleado para el desarrollo del proyecto desde la primera presentación de la idea de proyecto por parte del tutor D. Hipólito Guzmán Miranda, hasta los trabajos previos a la defensa del mismo.

En primer lugar, allá por el mes de noviembre del año 2015, se produjo la primera toma de contacto con lo que iba a ser el proyecto. En una primera tutoría con el profesor D. Hipólito Guzmán Miranda, éste expone la idea que tiene para la realización de un trazador de requisitos minimalista para diseños HDL ya que, sería una herramienta bastante útil para el equipo de trabajo del que forma parte. Una vez expuesta esta idea y esbozada lo que sería una solución deseable, se pasa a hacer un estudio del estado del arte para ver, por un lado, las diferentes opciones que existen en el mercado, tanto de software libre como de pago, y por otro, si de verdad merece la pena embarcarse en un proyecto de este tipo.

Durante este estudio del estado del arte, en primer lugar se hace una pequeña toma de contacto con todo lo referente a la Ingeniería de Requisitos y la gestión de requisitos, se estudia el significado de los diferentes términos empleados en estas disciplinas, en qué consisten, que información aportan y, como sería su implementación software para poder generarlos. Una vez realizada esta primera toma de contacto, sabiendo ya en que mundo se mueve el proyecto y teniendo las ideas claras de lo que necesitaría la aplicación para cumplir con las características deseadas por el tutor, se pasa a la realización de una búsqueda y estudio de las aplicaciones disponibles que pueden dar solución al problema planteado. Por un lado se buscan aplicaciones de software libre de las que se saca la conclusión de que para diseños HDL no existe ninguna específica y, por otro lado, se buscan aplicaciones bajo licencia de pago, de las que se ve que hay una gran cantidad para la gestión de requisitos y proyectos en general pero que, específicas para proyectos HDL solo se encuentran 2, *Spec-Tracer* de la compañía *Aldec* y *ReqTracer,* de la compañía *Mentor.* Se descubre una tercera aplicación, que aunque no es específica para diseños HDL, sí que se puede adaptar, se trata de *IBM Rational DOORS,* de la compañía *IBM*. Se pasa a investigar sobre las aplicaciones que proporcionan *Aldec* y *Mentor*, intentando

descargar versiones de prueba pero en ambos casos, cuando procedían a la validación de los datos solicitados, o nos llevaban a la versión de estudiantes, la cual no trae nada referente a "requirements management" o directamente denegaban la solicitud, por lo que se tuvo que ver la funcionalidad del programa, estudiando el datasheet y viendo vídeos de usuarios que si tenía la versión que se necesitaba y hacían una explicación del funcionamiento de la misma.

Por último, una vez acabada la fase de desarrollo, se pasa a una última fase de pruebas sobre diferentes proyectos y diversas situaciones particulares.

4.2 Tecnologías utilizadas

En esta sección se describen las tecnologías utilizadas para la implementación del trazador de requisitos y el motivo por el cual se ha elegido ese tipo de tecnología y no otra. A continuación se enumeran las tecnologías utilizadas.

1. **Python:** Se ha utilizado este lenguaje de programación para la parte de script ya que no necesita de compilador para la ejecución del código generado y, debido a que una de las condiciones era un sistema minimalista que no necesitase muchos recursos externos para funcionar se eligió este lenguaje para el script. Frente a otros lenguajes de script como pueden ser *Bash* o *Dash* se ha preferido el uso de *Python* debido a la gran diversidad de funciones que tiene implementadas para el tipo de código que se quería hacer. Entre dichas funciones se pueden destacar la librería *csv,* muy utilizada durante el proyecto para la apertura de los archivos de requisitos y la generación del archivo de salida o también, la ventaja que tiene el uso de diccionarios de *Python* para el manejo de datos. Otro recurso muy usado durante el proyecto fueron las listas y operaciones con las mismas, campo en el que el uso de este lenguaje otorgaba muchas ventajas. Además de todo esto, *Python* tiene una depuración que resulta ser mucho más fácil en contraposición a lenguajes como *Bash* o *Dash.*

Figura 4.1 Logo *Python*.

2. **HTML:** Se ha utilizado este lenguaje de etiquetas para poder generar la página web estática solicitada en la que se muestra el estado del proyecto. La opción no ha sido una elección ya que se especifica en los requisitos que se debía generar una página *HTML* para presentar los resultados.

Figura 4.2 Logo *HTML-5*.

3. **CSS:** Este lenguaje se ha usado para crear la hoja de estilos para dar forma a la página *HTML.*

Figura 4.3 Logo *CSS-3*.

4. **JavaScript:** Este lenguaje se ha usado para la generación de contenido dinámico en la página *HTML* ya que en función del archivo generado con el script *Python*, el contenido debía de ser distinto en cada caso. La elección del lenguaje *JavaScript* frente a otros como pueden ser, entre otros, *PHP,* ha sido nuevamente debido a la condición de minimalista solicitada ya que no se quería que se dependiese de ningún tipo de servidor web para la interpretación del código a la hora de generar la página *HTML.* Por ello se seleccionó uno que pudiese ser interpretado en local sin necesidad de ningún tipo de servidor ni en la máquina local ni en ninguna remota.

Figura 4.4 Logo *JavaScript*.

4.3 Desarrollo del proyecto

En esta sección se describe paso a paso como se ha realizado todo el proyecto, además de una descripción de su funcionamiento, pero sin entrar en detalle en el funcionamiento de cada bloque ya que eso se realizará en secciones posteriores.

A la vista del resultado obtenidos en el estudio del estado del arte, se comenzó la etapa de desarrollo del proyecto la cual, se dividió en cuatro fases. La primera se dedicó al desarrollo de un script el cual, buscando en los archivos del proyecto y comparándolos con los archivos de requisitos pudiese generar un fichero *csv* a modo de informe en el cual estuviesen reflejados todos los requisitos encontrados, su estado de implementación, los tests encontrados, si se habían pasado con éxito o habían generado algún error, los archivos en los cuales se encontraban los requisitos y además, se generaban también dos matrices de trazabilidad.

4.3.1 Fase 1. Script

Para esta primera parte se usa lenguaje de script *Python.* Se comienza con la realización de unos diagramas de estado en los que se dibuja el comportamiento completo del programa y después, unos diagramas de flujo en los cuales se representa el comportamiento de cada función. Estos diagramas de funcionamiento se pueden ver en la sección *"Arquitectura del Software"* en la parte correspondiente al "Script" Se usan pequeños scripts independientes para testear las funciones por separado, una vez dichas funciones hacen lo deseado se unen en un fichero independiente hasta conseguir tener el código completo. Con el código completo, se realiza una depuración del mismo en la cual se crea una librería para todas las funciones implementadas dejando la función principal lo más reducida posible pero a la vez entendible por el usuario el cual puede saber qué hace en cada paso el programa tan solo leyendo la función principal.

4.3.2 Fase 2. Test del Script

La segunda fase se dedica a la realización de pruebas sobre el script implementado intentando abarcar el mayor número de posibilidades que se pueden presentar en un proyecto real en base a los requisitos. Además de estas pruebas, se utilizan proyectos reales como son *Edelweiss* y el propio *Minitracer* a los cuales se les

aplica el trazador de requisitos para ver su correcto funcionamiento.

4.3.3 Fase 3. Interfaz Web

En cuanto a la tercera fase, esta se corresponde con el desarrollo de una aplicación web. Dicha aplicación se puede dividir en dos partes:

1. Una página web con un formulario para introducir los requisitos del proyecto y generar el archivo *csv* que posteriormente será utilizado como una de las entradas del script para la generación de los resultados.

2. Una página web donde se presentan los resultados obtenidos del script. Esta interfaz consta de los siguientes detalles:

   - Se usa la librería de *JavaScript* denominada *Papaparse* para la lectura del archivo de resultados. Ésta recibe un objeto de tipo *"File"* y genera otro objeto con los resultados. En cuanto a la licencia de la nombrada librería, se trata de una licencia de software libre el cual se puede usar por cualquier usuario y para cualquier tipo de aplicación.

   - Se utiliza también una plantilla de estilos *CSS* para la página web. Al igual que en el caso anterior, esta plantilla de estilos también cuenta con una licencia de software libre que permite su uso para este proyecto y que además, permite hacer los cambios necesarios sobre la misma.

   - La interfaz web cuenta con una página de inicio en la que el usuario puede elegir si lo que desea es generar los resultados de un proyecto o, de lo contrario, desea generar un archivo *csv* con requisitos.

Una descripción más detallada de esta parte del software se realiza en secciones posteriores.

4.3.4 Fase 4. Tests

La cuarta fase se centra en el testeo de la aplicación completa, la fase consta de dos partes:

1. La primera consta de la realización de los mismos tests que en la *"Fase 2"* pero esta vez se ven los resultados en la página *HTML.*

2. La segunta consiste en la aplicación del proyecto *minitracer* sobre dos proyectos reales, por un lado el proyecto *edelweiss,* proyecto de excelencia de la Junta de Andalucía, perteneciente al departamento de Ingeniería Electrónica de la Universidad de Sevilla y que consiste en el diseño de un sistema de comunicaciones inalámbrico para aplicaciones aeroespaciales[33]. De este proyecto solo se realizan pruebas sobre la parte del transmisor. Por último, también se aplica el software realizado a sí mismo como última prueba de comprobación de su correcto funcionamiento.

4.4 Arquitectura del Software

En lo que corresponde a la aplicación en sí, su esqueleto se puede observar en la Figura 4.5. Éste se compone de 2 cajas negras, la primera correspondería a un script programado en *Python*, el cual tendría como entrada 3 elementos. El primero sería la ruta a un archivo *.txt* con las rutas de los ficheros csv que almacenan los requisitos del proyecto, el segundo, la ruta a otro archivo *.txt* con las rutas a los archivos fuente de nuestro proyecto, y por último, un tercer elemento que sería la ruta a la carpeta del proyecto. La salida de esta primera caja sería un archivo *.csv* con los resultados del estado de nuestro proyecto. La segunda caja negra se corresponde con una aplicación web en la cual se interpreta el archivo *.csv* generado en el script. Dicha aplicación consiste en una página web en la que se puede ver tanto un resumen del estado el proyecto, como una descripción detallada del estado del mismo además de, dos matrices de trazabilidad. La primera mostraría requisitos frente a tests, y la segunda requisitos frente a los archivos fuente del proyecto donde se encuentran.

Como funcionalidad extra también se añade, dentro de la aplicación web, una ventana para poder generar los ficheros *.csv* con los requisitos del proyecto.

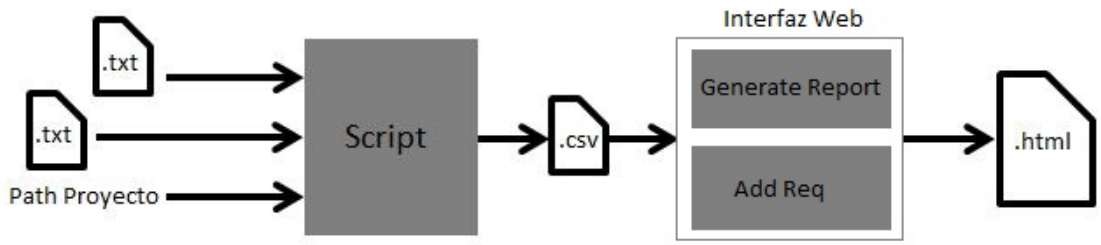


Figura 4.5 Modelo *Black-box*.

En los siguiente apartados pasamos a explicar de manera detallada cada una de las cajas del modelo *Black-Box*

4.4.1 Archivos de entrada

La primera parte de este software correspondería a los archivos necesario a la entrada del script. Además del *path* del proyecto bajo estudio, los mencionados archivos se corresponderían con 2 ficheros de texto plano en los que se especifican las rutas a cada fichero del sistema. En primer lugar un archivo de texto plano llamado en el que están guardadas las rutas a los ficheros *csv* con los requisitos, y en segundo lugar otro archivo de texto plano con las rutas a los archivos de código fuente que contiene el proyecto o, al menos, los archivos en los que se encuentren implementados los requisitos. En las figuras se pueden ver un ejemplo de este tipo de archivos, la Figura 4.6 se corresponde con el fichero de texto plano con las rutas a los archivos de requisitos y la Figura 4.7 se corresponde con el que contiene los *paths* de los ficheros de código.

Figura 4.6 Archivo rutas requisitos.

Figura 4.7 Archivo rutas ficheros de código.

4.4.2 Archivo de requisitos

Este archivo se genera a través de la interfaz web, concretamente en la pestaña de *add requirements* y sirve para recoger los requisitos de los que consta el proyecto. Por cada requisito se piden varios parámetros a introducir que, a pesar de no ser obligatorios para el correcto funcionamiento del software, si que son recomendables para su mejor entendimiento en revisiones futuras. No obstante, se trabaja para versiones futuras en un algoritmo que permita procesar el formulario de requisitos y genere errores si hay campos necesarios a los que no se les ha dado valor. En la Figura 4.8 se puede ver un ejemplo de archivo *csv* generado a través de la aplicación web.

	A	B	C	D	E
1	PROJECT NAME	Minitracer			
2	LABEL	TYPE	PARAMETER	DESCRIPTION	TEST
3	Gestion_req	Ctest		Gestión de los requisitos comparando los encontrados en csv y en archivos de proyecto	MINITRACER_PASS
4	Gestion_test	Ctest		Comparación entre los resultados de los test generados y los test encontrados en csv	MINITRACER_PASS
5	Error_test_reservado	Ctest		Ningún test del proyecto se puede llamar como un test reservado de minitracer	MINITRACER_PASS
6	Escribe_nombre_proyecto	Ctest		Se escribe en fichero de salida el nombre del proyecto	MINITRACER_PASS
7	Escribe_resultado_reg	Ctest		Se escribe en fichero de salida el resultado de procesar los requisitos	MINITRACER_PASS
8	Escribe_resultado_test	Ctest		Se escribe en archivo de salida el resultado de procesar los tests	MINITRACER_PASS
9	Escribe_matrices	Ctest		Se escriben en el fichero de salida las matrices de trazabilidad	MINITRACER_PASS
10	Test_reservados	Ctest		Existen test reservados para el minitracer	MINITRACER_PASS
11	Arg_linea_comand	Ctest		Se leen los argumentos de linea de comandos	MINITRACER_PASS
12	Fichero_path_csv	Ctest		Los ficheros de requisitos deben tener sus paths escrito en un archivo txt	MINITRACER_PASS
13	Fichero_path_src	Ctest		Los ficheros del proyecto deben tener sus paths escrito en un archivo txt	MINITRACER_PASS
14	Busca_req_csv	Ctest		Existe función que busca los requisitos en los archivos csv	MINITRACER_PASS
15	Mismo_project	Ctest		Existe función que compara que los csv pertenezcan al mismo proyecto	MINITRACER_PASS
16	Busca_req_src	Ctest		Función que busque requisitos en ficheros del proyecto	MINITRACER_PASS
17	Compara_resultados	Ctest		Se comparan los requisitos de los csv y los ficheros del proyecto	MINITRACER_PASS
18	Consulta_test	Ctest		Consulta test generados con CTest	MINITRACER_PASS
19	Calculo_estado_proyecto	Ctest		Se calcula el estado del proyecto	MINITRACER_PASS
20	Matriz_req_path	Ctest		Existe una matriz de trazabilidad requisitos-path	MINITRACER_PASS
21	Matriz_req_test	Ctest		Existe una matriz de trazabilidad requisitos_test	MINITRACER_PASS
22	Genera_csv_out	Ctest		Se genera un fichero csv con los datos del estado del proyecto	MINITRACER_PASS
23	Gestion_error_input	Ctest		Si se produce un error con los argumentos en la linea de comandos se muestra el modo correcto	MINITRACER_PASS
24					

Figura 4.8 Archivo de requisitos.

4.4.3 Script

Como se ha podido ver en la Figura 4.5 el modelo correspondiente a esta parte del programa se corresponde de un script con 3 entradas y una salida. Pasamos a explicar su funcionamiento y a detallar las entradas una a una.

El script programado en *Python* realizado consta de 3 entradas, una primera es un archivo de extensión *txt* en el cual se recogen las rutas de todos los archivos *csv* que contienen los requisitos de nuestro sistema. Este archivo estará compuesto por tantas líneas como ficheros *csv* existan. Del mismo modo, la segunda entrada al programa será otro archivo *txt* en el cual se recogen todas las rutas de los archivos de código implicados en el proyecto. Estos pueden ser todos los que compongan el proyecto o solo los que tengan requisitos. Se deja al usuario libre elección para introducir los ficheros que estime oportunos. A pesar de que el trazador de requisitos se ha realizado para proyectos HDL, en este archivo se pueden introducir ficheros con cualquier tipo de extensión siempre y cuando sean ficheros de texto y no binarios y el usuario tenga permiso de apertura y lectura. Por último, la tercera entrada corresponde con el path de la carpeta *build* (o similar) contenida dentro del proyecto, es decir, la carpeta donde el programa espera encontrar los ficheros generados por *CMake* y *CTest*. Esta dirección servirá, tanto para ubicar los ficheros de salida del trazador como para poder encontrar los archivos con los resultados de los tests pasados por *CTest.* En cuanto a la ejecución de este script, este se ejecutará a través de la línea de comandos con el formato que se puede ver en la Figura 4.9.

```
carlos@carlos:~$
carlos@carlos:~$ ./minitracer.py ~/devel/edelweiss/requisitos.txt ~/devel/edelweiss/source.txt ~/devel/edelweiss/build/
```

Figura 4.9 Ejecución script.

Aunque también se podría ver su formato de ejecución introduciendo solo el nombre del ejecutable del trazador y pulsando la tecla *enter,* en cuyo caso aparecería un error indicando cómo debe de ejecutarse el código. Dicho ejemplo puede verse en la Figura 4.10.

```
carlos@carlos:~/devel/minitracer/minitracer/script$ ./minitracer.py

Error on input arguments
Input example:
        ./<PROGRAM_NAME> <FILE_WITH_REQ_PATHS> <FILE_WITH_SRC_PATHS> <PROJECT_PATH>
```

Figura 4.10 Información sobre ejecución de script.

Una vez ejecutado el código su funcionamiento se resume en lo siguiente:

1. Se abre el archivo con las rutas de los ficheros *csv* con los requisitos y se guardan en una lista.
2. Se procede del mismo modo con los ficheros de código.
3. Se procede a abrir los ficheros *csv* y leer el contenido creando diccionarios.
4. Se abren los archivos de códigos buscando la cadena *@Req <MINITRACER_PASS>* para encontrar los requisitos.
5. Se comparan los requisitos obtenidos de los ficheros de código con los existentes recogidos de los archivos *csv.*
6. Se buscan los resultados de los tests, generados éstos por la herramienta *CTest.*
7. Se comparan esos resultados con los tests recogidos de los ficheros csv.
8. Se crean las matrices tanto de trazabilidad como de verificación.

9. Se calculan el número de requisitos y tests totales del proyecto y con esto se generan los porcentajes de desarrollo del proyecto.

10. Se escribe el archivo *csv* que será la salida del script. Este archivo estará ubicado en la carpeta *build* del proyecto en cuestión.

En Figura 4.11 se puede ver un diagrama de estados que muestra un resumen del funcionamiento antes mencionado.

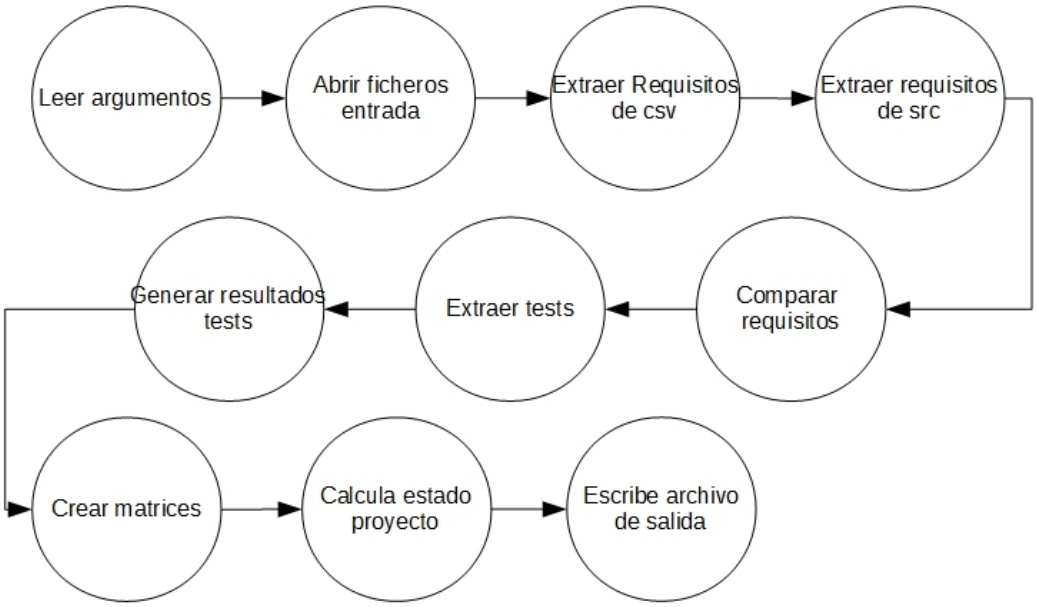


Figura 4.11 Diagrama de estados Script.

En cuanto a la salida de errores, si durante la ejecución del código se produjese algún error referente a permisos de lectura y/o escritura del alguno de los archivos pasados o generados, o bien no se encontrase alguno de los archivos especificados, el programa interrumpiría su ejecución informando de que ha habido un error al abrir el archivo en cuestión.

En lo que se refiere al código, este consta de una función *main* cuyo contenido se puede ver en el siguiente fragmento:

Código 4.1 Función main.py del trazador de requisitos.

```
if __name__ == '__main__':

    try:
        #Declaracion de variables (listas)
      ...
        #Declaracion variables enteras
      ...

        test_reservados = {'MINITRACER_MANUAL' : 0 , 'MINITRACER_PASS' : 1 ,'
            MINITRACER_FAIL' : 0}
```

```
    archivos_csv = sys.argv[1]
    archivos_src = sys.argv[2]
    project_path = sys.argv[3]

    if archivos_csv == archivos_src:
        raise IndexError ('Invalid input')

    archivo_resultados = project_path+'result_req.csv'

    test_file_path = project_path+"Testing/Temporary/CTestCostData.txt"

    paths_csv = open_path_files(archivos_csv)
    paths_src = open_path_files(archivos_src)

    num_paths_csv = len(paths_csv)
    num_paths_src = len(paths_src)

    reqcsv, typecsv, paramcsv, testcsv, project_namecsv = open_csv_files(
        paths_csv)
    sameProjectName = sameProjectName(project_namecsv)

    reqsrc = open_src_files(paths_src)

    compared_csv,compared_src,compared_pathInCsv,compared_pathNotInCsv,
        cont_porcentaje_req = compare(reqcsv, reqsrc)

    result_test, result_test_missing, result_test_missing_project,
        number_test, porcentaje_test, status_tests = search_test(
        test_file_path,testcsv,test_reservados)

    number_req = len(reqcsv)+len(compared_src)
    number_global = number_req + number_test
    porcentaje_req = (cont_porcentaje_req*100)/number_req
    porcentaje_global = ((cont_porcentaje_req+((porcentaje_test*number_test)
        /100))*100)/number_global

    traceability_matrix = gen_matrix(reqsrc, paths_src)

    test_matrix = gen_matrix(testcsv, result_test.keys())

    write_file(archivo_resultados, project_namecsv, reqcsv, typecsv,
        paramcsv, testcsv,compared_csv, compared_src, compared_pathInCsv,
        compared_pathNotInCsv, result_test, result_test_missing,
        result_test_missing_project, traceability_matrix, test_matrix,
        number_test,number_req,number_global, porcentaje_global,
        porcentaje_req,porcentaje_test, status_tests, num_paths_csv,
        num_paths_src)

except IndexError:
    print "\nError on input arguments"
    print "Input example: \n\t ./[PROGRAM_NAME] [FILE_WITH_REQ_PATHS] [
        FILE_WITH_SRC_PATHS] [PROJECT_PATH]\n"
except:
    print("Unexpected error: ", sys.exc_info()[0])
    raise
```

4.4.4 Interfaz Web

Con respecto a la *Interfaz Web,* el modelo de caja negra mostrado antes, descompuesto en dos sub-módulos que, aunque formen parte del mismo fichero *HTML,* una parte sería el generador de ficheros *csv* con los requisitos y la otra parte sería la encargada de la representación del estado del proyecto. A continuación pasamos a describir cada parte por separado.

En primer lugar nos vamos a centrar en la parte de la aplicación web correspondiente a la pantalla de inicio. En ella se puede elegir si lo que se desea es añadir requisitos para generar el fichero *csv* o bien adjuntar el fichero de resultados para generar la página que muestra el estado del proyecto. En la Figura 4.12 se puede ver dicha pantalla.

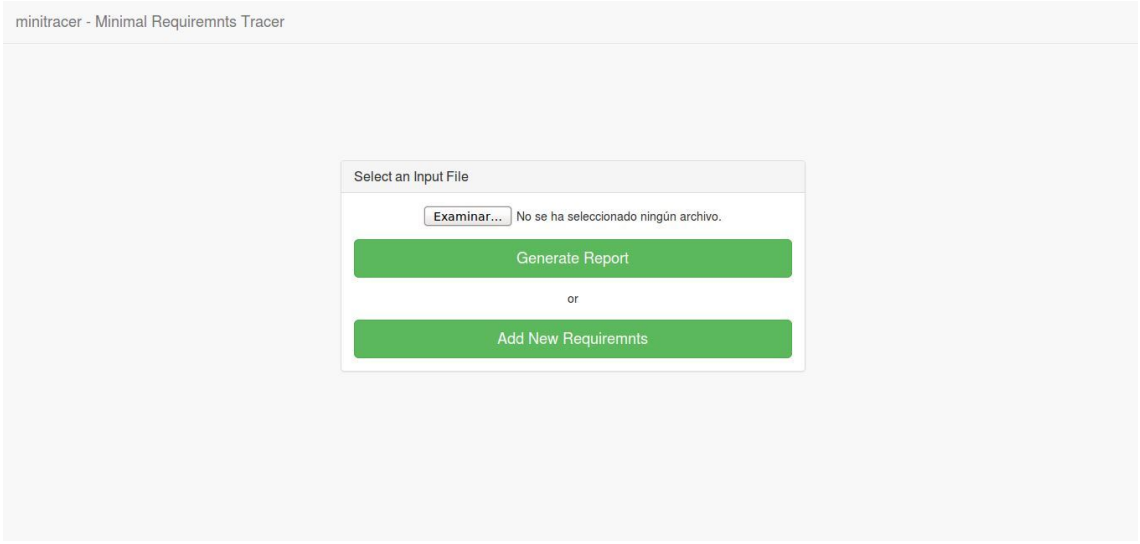


Figura 4.12 Página de inicio Interfaz Web.

Si seleccionamos la opción *"Add New Requirements"*, la aplicación nos redirigirá a la página mostrada en la Figura 4.13 en la que podremos ver un formulario en el que podremos ir anotando los requisitos añadiendo filas para nuevos requisitos en caso de que fuese necesario.

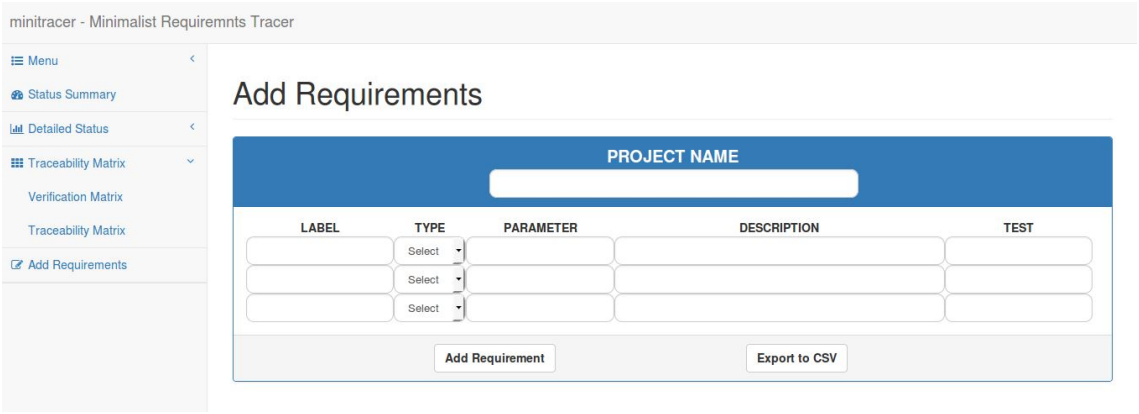


Figura 4.13 Página para añadir requisitos.

La segunda parte de esta interfaz web correspondería a la parte de presentación de los resultados, para acceder a la misma, desde la misma página de inicio se selecciona *"Generate Report"* y lo primero que se pide antes de redireccionarnos a la página de resultados es que indiquemos el archivo *csv* que se generó en el script. En la Figura 4.14 se puede ver esta parte del proceso.

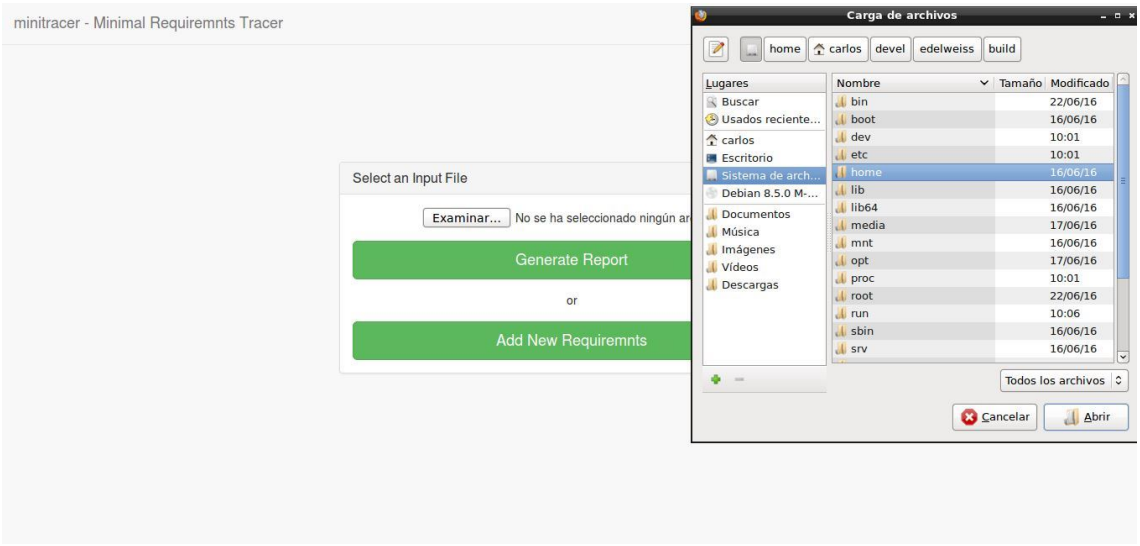


Figura 4.14 Selección de archivo de resultados.

Una vez seleccionado el archivo deseado y haciendo click sobre *"Generate Report,"* se redireccionaría al usuario a la página con el resumen del proyecto desde la que, podría ir accediendo a las diferentes partes de la interfaz a través del menú. En la Figura 4.15 se puede ver la página con el resumen del estado del proyecto. En primer lugar se pueden ver unos indicadores con el número de requisitos, tests, archivos de código y archivos de requisitos con los que cuenta nuestro proyecto e, inmediatamente debajo de ellos, unas barras de progreso con el estado global, del los requisitos, y de los tests. Estas barras irán cambiando de color en función del estado de progreso en el que se encuentre el proyecto.

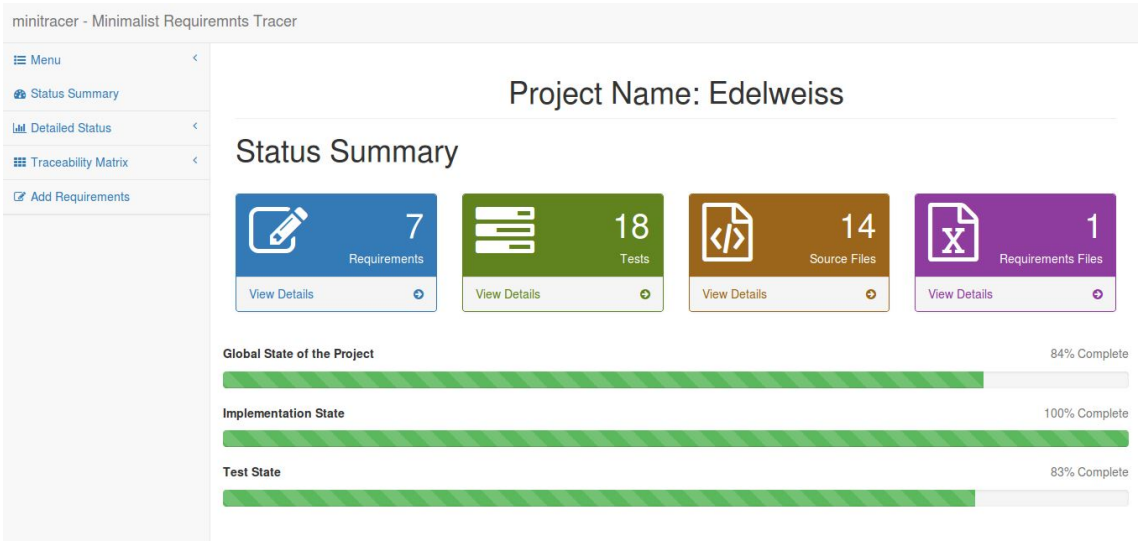


Figura 4.15 Página resumen de resultados.

Si navegamos por el menú lo primero que encontramos en la opción *"Detailed Status"* la cual es un desplegable que nos permite el acceso a 3 páginas diferentes. En el primer caso nos encontramos con la página *"Detailed Requirement Status"* en la que podemos encontrar una tabla en la que aparecen todos los requisitos, algunos detalles, los tests asociados, el número de tests que pasan y si se encuentran o no tanto en el fichero *csv* como el el proyecto. Como se puede apreciar en la imagen, a la columna con el número de test pasados se le ha asociado un código de colores asignándose el color rojo si no se ha pasado ningún test, el naranja si se pasan algunos y el verde en caso de que se pasen todos. Además, como funcionalidad extra, en la tabla se encuentra la opción de poder buscar por algún parámetro en concreto u ordenarla en función del parámetro que el usuario desee. En la Figura 4.16 se muestra un ejemplo.

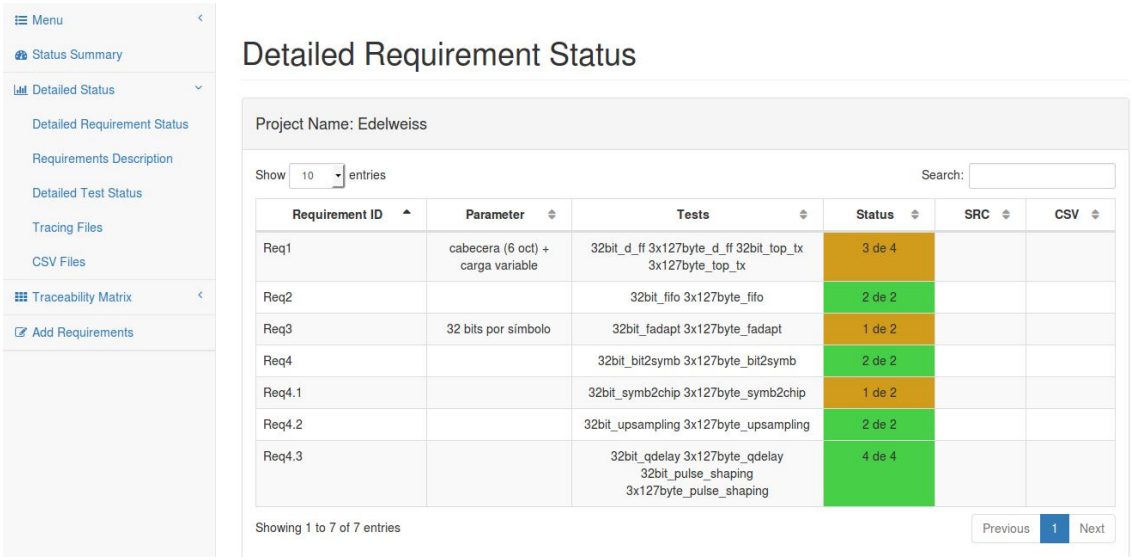


Figura 4.16 Página con detalles de requisitos.

La segunda opción corresponde a una descripción de cada requisito. En esta ventana podemos encontrar una tabla en la que se muestran el ID del requisito y a continuación su descripción. En la Figura 4.17 se puede observar un ejemplo.

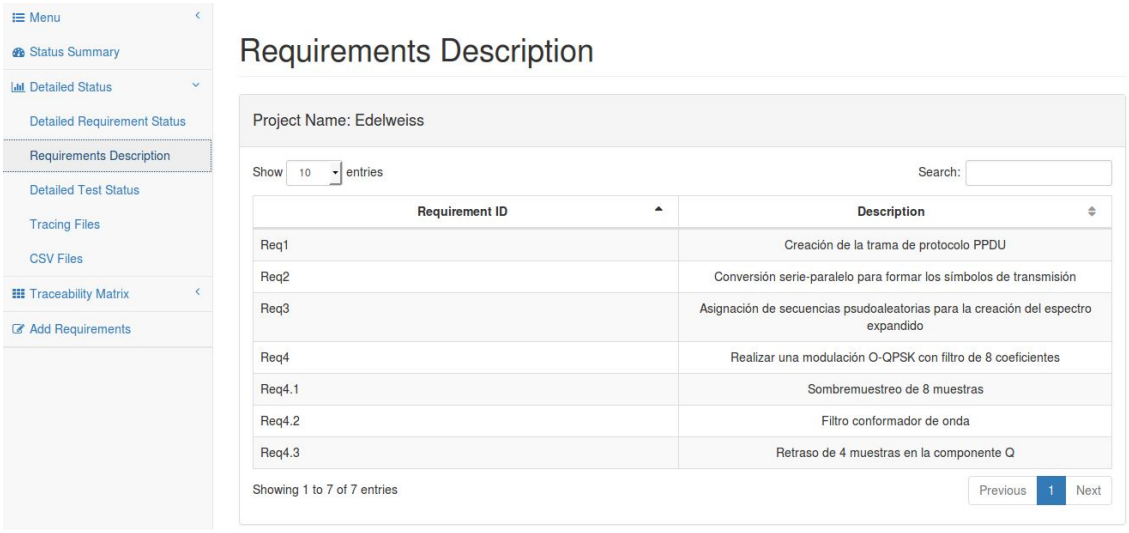


Figura 4.17 Página con descripción de requisitos.

La tercera opción de este desplegable es *"Detailed Test Status"*, esta página sigue la misma tónica que la anterior, una tabla con los datos y el buscador pero en este caso, los datos que encontramos es del estado en el que se encuentran los tests del proyecto así como si aparecen tanto en el csv como en el proyecto o de lo contrario falta en alguno de los dos. En cuanto al código de colores asignado a la columna del estado se le asigna el color verse si se encuentra el test y además se pasa, el naranja si el test ha sido encontrado pero no se pasa y el color rojo acompañado de la palabra *ERROR* en caso de que el test no se encuentre en el proyecto. En la Figura 4.18 se muestra un ejemplo.

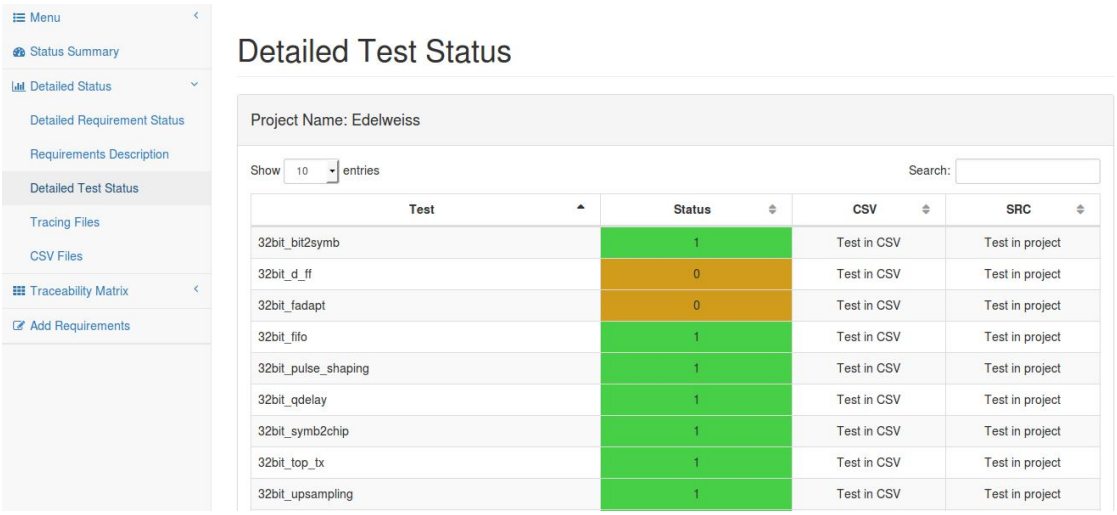


Figura 4.18 Página con detalles de tests.

En la cuarta opción de esta parte del menú nos encontramos con la pestaña *"Tracing Files"*, en ella se nos muestra nuevamente una tabla cuyo con tenido son los requisitos encontrados y los archivos del proyecto en los que se encuentran. En caso de que el requisito no se encontrase en el proyecto también se indicaría. En la Figura 4.19 se puede ver dicha página.

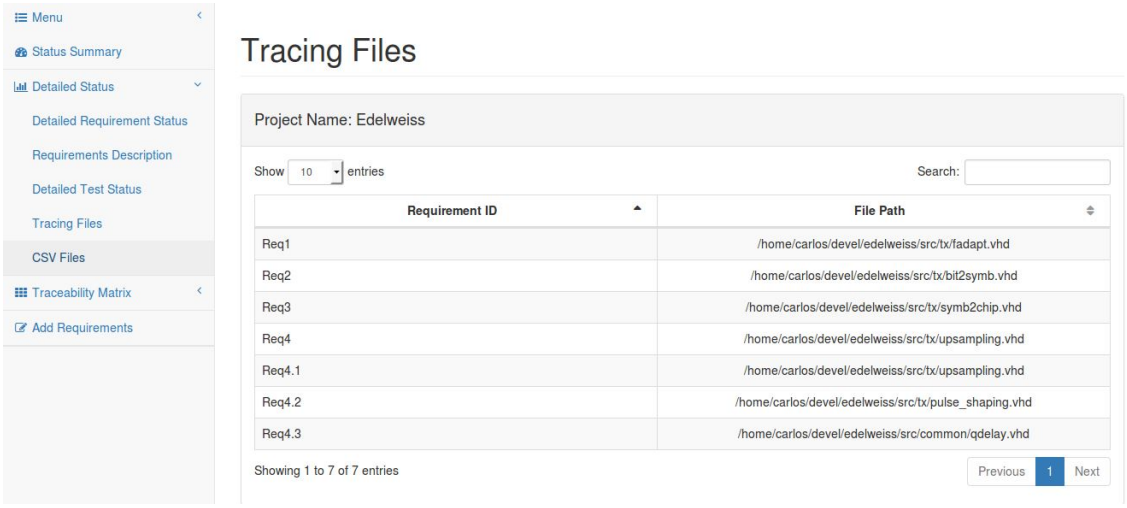


Figura 4.19 Página con los ficheros del proyecto.

Como quinta opción de este sub-menú, encontrado una pestaña referente a los archivos de requisitos. En esta se encuentran los tests y el fichero de requisitos *csv* en el que se encuentran. En la Figura 4.20 se puede ver un ejemplo.

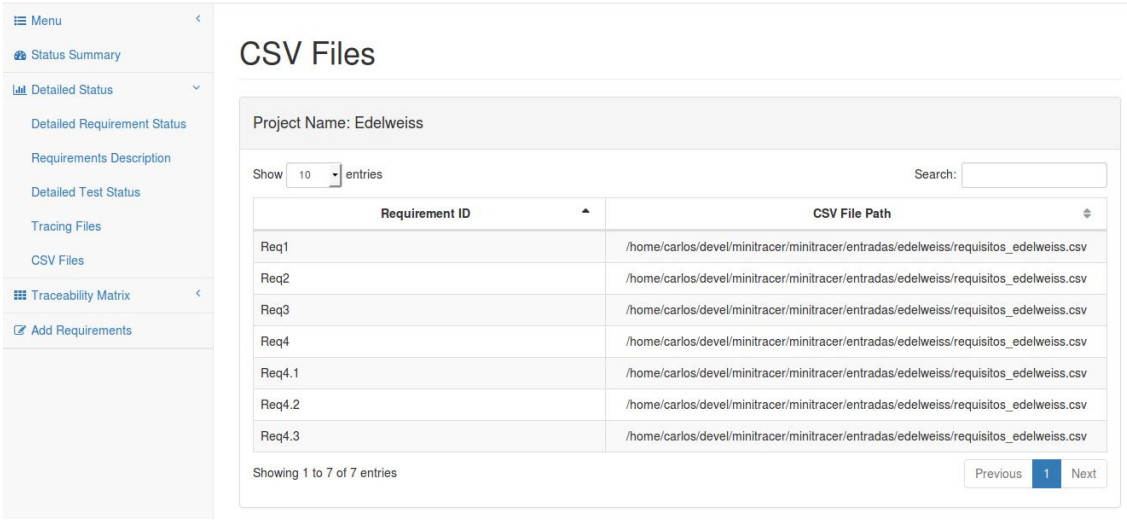


Figura 4.20 Página con los ficheros de requisitos del proyecto.

Continuando con la navegación sobre el menú, la siguiente pestaña que se encuentra es *"Traceability Matrix"*, en ella se despliegan dos sub-pestañas, en la primera se puede observar una matriz en la que se muestran requisitos frente a test (Figura 4.21) y en la segunda una nueva matriz en la que esta vez se representan requisitos frente a paths (Figura 4.22), en ambas para hacer más fácil su localización, las celdas que marcan las correspondencias con una "X", además se señalan en verde.

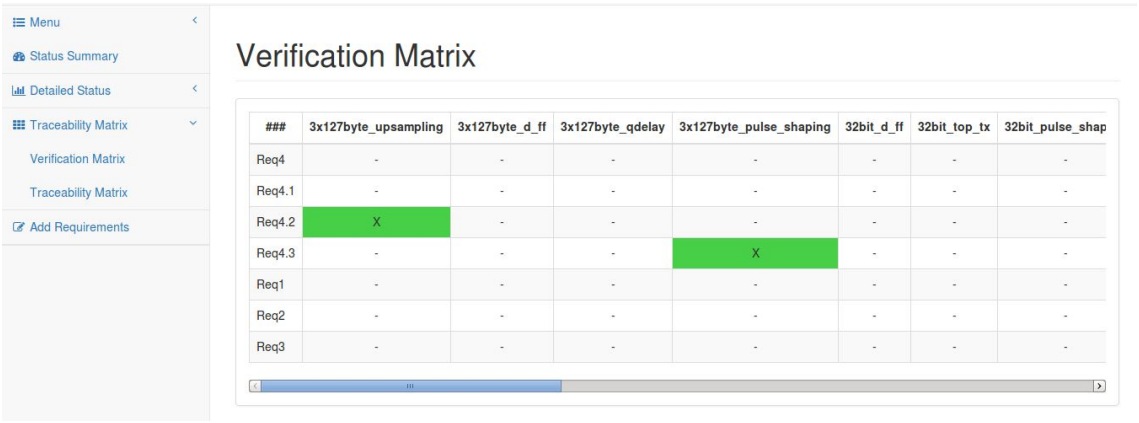


Figura 4.21 Página con matriz de verificación.

Figura 4.22 Página con matriz de trazabilidad.

5 Resultados

Un resultado no es un resultado si no cumple con los requisitos impuestos

ANÓNIMO

En el siguiente capítulo se habla acerca de los resultados obtenidos una vez finalizado el proyecto y sometido a varias pruebas como son la aplicación a otros proyectos y la generación de requisitos y tests de prueba intentando abarcar todas las posibilidades que se pueden encontrar en el desarrollo de un proyecto.

5.1 Introducción a los resultados

Una vez realizado el proyecto se aplicó a dos proyectos para comprobar, por un lado, su correcto funcionamiento, y por otro que el archivo *HTML* generado cumplía con las expectativas solicitadas. Los proyectos a los que se aplicó dicha aplicación fueron, por un lado el proyecto *edelweiss,* proyecto perteneciente al departamento de Ingeniería Electrónica, y por otro lado se aplicó el trazador de requisitos al propio software del trazador de requisitos o *minitracer* como también se ha llamado. Además de todo esto, el proyecto se sometió a una gran variedad de situaciones que se pueden encontrar en un proyecto, este tipo de situaciones particulares son:

1. Que existan requisitos en el fichero *csv* que no se encuentren en los ficheros del proyecto.
2. Que existan requisitos en los ficheros del proyecto que no se encuentren recogidos en el fichero. *csv.*
3. Que existan tests en el fichero *.csv* que no se encuentren en el proyecto.
4. Que existan tests en el proyecto que no se encuentren recogidos en el fichero *csv.*
5. Que se introduzcan ficheros de requisitos pertenecientes a proyectos distintos.
6. Que no se especifiquen correctamente las entradas del script.
7. Que alguno de los archivos especificados no exista.
8. Que la carpeta *build* (o similar) no exista en nuestro proyecto ya que no se han ejecutado con anterioridad las ordenes *CMake* y *CTest.*

En las siguientes secciones se pueden encontrar los resultados de aplicar el software *minitracer* a los diferentes proyectos mencionados y a las pruebas descritas.

5.2 Minitracer aplicado al proyecto Edelweiss

En la aplicación al proyecto *Edelweiss*[1], los resultados obtenidos se pueden ver en las siguientes imágenes. Comenzando por la Figura 5.1 en ella se puede ver el resumen del estado del proyecto en la que se facilitan el número de requisitos existentes, el número de test, los ficheros de código existentes y los ficheros de requisitos existentes. Inmediatamente debajo se pueden ver las barras de progreso, observando como la referente al estado de los requisitos está al completo mientras que la de los test no ya que existen test que han dado fallo a la salida.

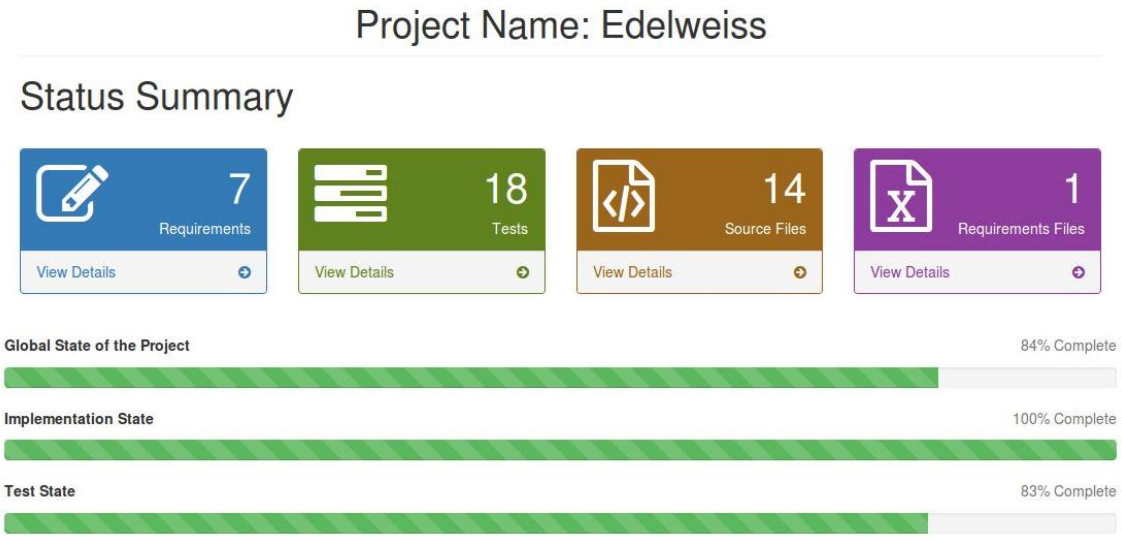


Figura 5.1 Status Summary Edelweiss.

[1] La pruebas sobre dicho proyecto se han realizado únicamente sobre el transmisor

En la Figura 5.2 se puede ver con detalle el estado de cada requisito y los tests que pasan cada uno.

Project Name: Edelweiss

Show 10 entries Search:

Requirement ID	Parameter	Tests	Status	SRC	CSV
Req1	cabecera (6 oct) + carga variable	32bit_d_ff 3x127byte_d_ff 32bit_top_tx 3x127byte_top_tx	3 de 4		
Req2		32bit_fifo 3x127byte_fifo	2 de 2		
Req3	32 bits por símbolo	32bit_fadapt 3x127byte_fadapt	1 de 2		
Req4		32bit_bit2symb 3x127byte_bit2symb	2 de 2		
Req4.1		32bit_symb2chip 3x127byte_symb2chip	1 de 2		
Req4.2		32bit_upsampling 3x127byte_upsampling	2 de 2		
Req4.3		32bit_qdelay 3x127byte_qdelay 32bit_pulse_shaping 3x127byte_pulse_shaping	4 de 4		

Showing 1 to 7 of 7 entries Previous 1 Next

Figura 5.2 Requirements Status Edelweiss.

En la Figura 5.3 se muestra una descripción de cada test.

Project Name: Edelweiss

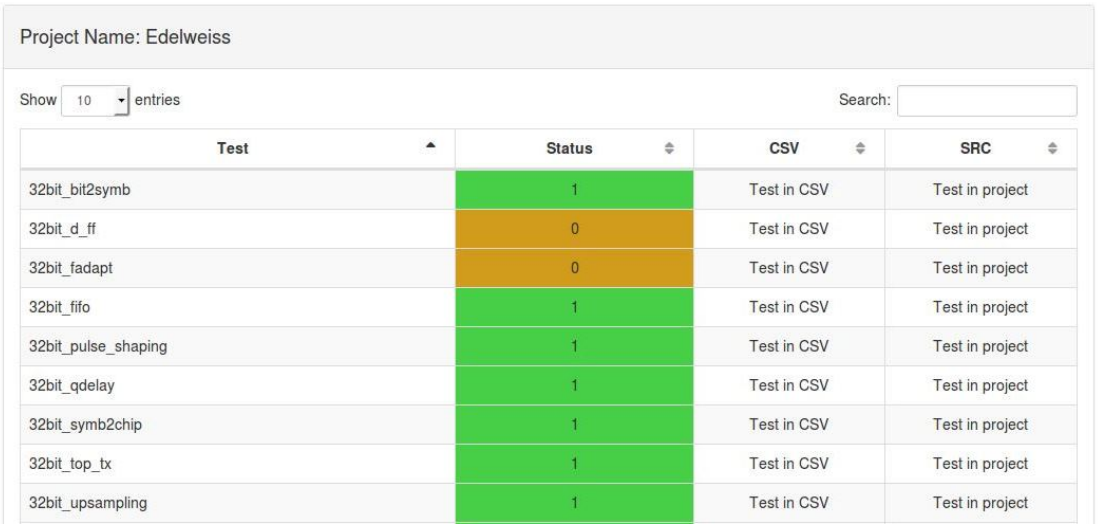
Show 10 entries Search:

Requirement ID	Description
Req1	Creación de la trama de protocolo PPDU
Req2	Conversión serie-paralelo para formar los símbolos de transmisión
Req3	Asignación de secuencias psudoaleatorias para la creación del espectro expandido
Req4	Realizar una modulación O-QPSK con filtro de 8 coeficientes
Req4.1	Sombremuestreo de 8 muestras
Req4.2	Filtro conformador de onda
Req4.3	Retraso de 4 muestras en la componente Q

Showing 1 to 7 of 7 entries Previous 1 Next

Figura 5.3 Requirements Description Edelweiss.

En la Figura 5.4 se muestra una descripción del estado de los tests.



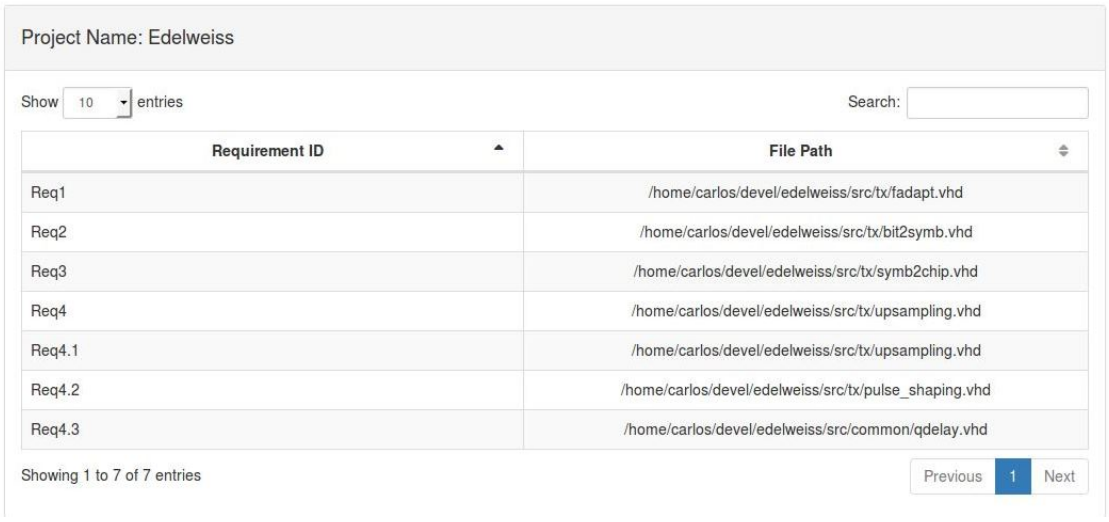
Project Name: Edelweiss

Show 10 entries Search:

Test	Status	CSV	SRC
32bit_bit2symb	1	Test in CSV	Test in project
32bit_d_ff	0	Test in CSV	Test in project
32bit_fadapt	0	Test in CSV	Test in project
32bit_fifo	1	Test in CSV	Test in project
32bit_pulse_shaping	1	Test in CSV	Test in project
32bit_qdelay	1	Test in CSV	Test in project
32bit_symb2chip	1	Test in CSV	Test in project
32bit_top_tx	1	Test in CSV	Test in project
32bit_upsampling	1	Test in CSV	Test in project

Figura 5.4 Tests Status Edelweiss.

En la Figura 5.5 se muestran los ficheros de código en los que se encuentra cada requisito.



Project Name: Edelweiss

Show 10 entries Search:

Requirement ID	File Path
Req1	/home/carlos/devel/edelweiss/src/tx/fadapt.vhd
Req2	/home/carlos/devel/edelweiss/src/tx/bit2symb.vhd
Req3	/home/carlos/devel/edelweiss/src/tx/symb2chip.vhd
Req4	/home/carlos/devel/edelweiss/src/tx/upsampling.vhd
Req4.1	/home/carlos/devel/edelweiss/src/tx/upsampling.vhd
Req4.2	/home/carlos/devel/edelweiss/src/tx/pulse_shaping.vhd
Req4.3	/home/carlos/devel/edelweiss/src/common/qdelay.vhd

Showing 1 to 7 of 7 entries Previous 1 Next

Figura 5.5 Tracing Files Edelweiss.

La Figura 5.6 muestra en su caso los ficheros **csv** en los que se encuentra cada requisito.

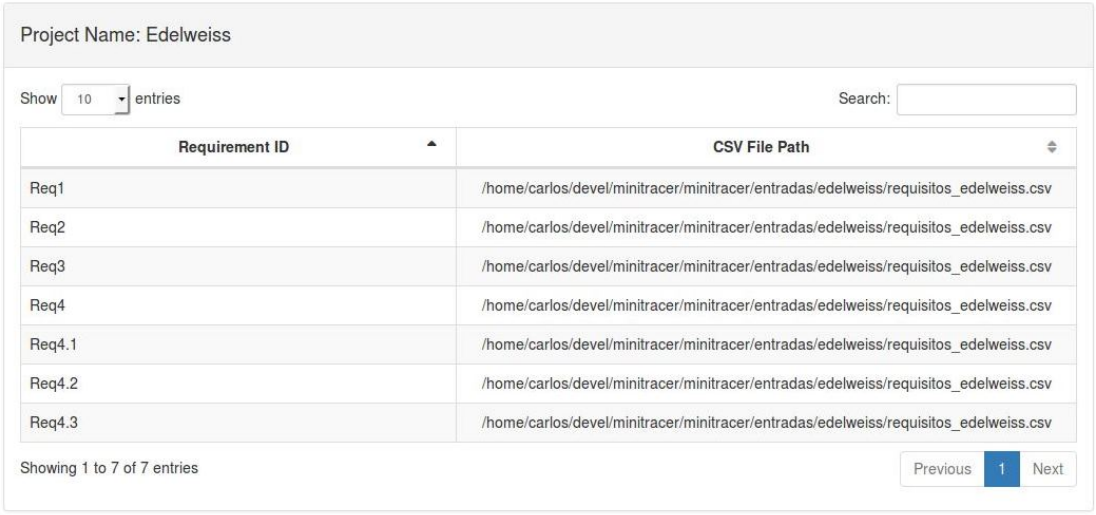
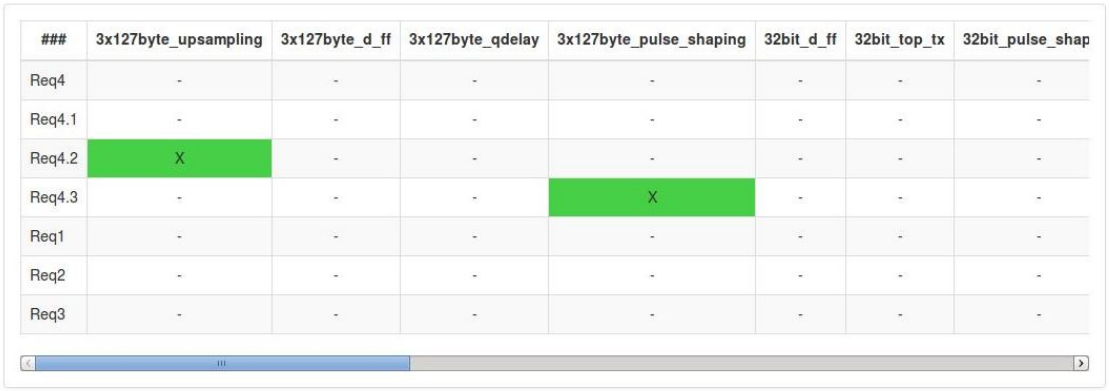


Figura 5.6 Requirements Files Edelweiss.

En la Figura 5.7 se muestra la matriz de trazabilidad que enfrenta requisitos y tests. Como se puede observar en esta matriz, los requisitos que están relacionados con los test aparecen marcados con una "X" mientras que el resto de elementos que no tienen relación aparecen con un "-".



###	3x127byte_upsampling	3x127byte_d_ff	3x127byte_qdelay	3x127byte_pulse_shaping	32bit_d_ff	32bit_top_tx	32bit_pulse_shap
Req4	-	-	-	-	-	-	-
Req4.1	-	-	-	-	-	-	-
Req4.2	X	-	-	-	-	-	-
Req4.3	-	-	-	X	-	-	-
Req1	-	-	-	-	-	-	-
Req2	-	-	-	-	-	-	-
Req3	-	-	-	-	-	-	-

Figura 5.7 Verification Matrix Edelweiss.

Por último en la Figura 5.8 se muestra la matriz de trazabilidad que enfrenta requisitos y ficheros. Como se puede observar en esta matriz, los requisitos que están relacionados con los archivos, es decir, los requisitos que están implementados dentro de un cierto archivo, aparecen marcados con una “X” mientras que el resto de elementos que no tienen relación aparecen con un “-”.

###	/home/carlos /devel /edelweiss /src/common /d_ff.vhd	/home/carlos/devel /edelweiss/src/common /edelweiss_common.vhd	/home/carlos /devel /edelweiss /src/common /qdelay.vhd	/home/carlos /devel /edelweiss /src/rx /cfilter.vhd	/home/carlos /devel /edelweiss /src/rx /dem_filter.vhd	/home/carlos/devel /edelweiss/src/rx /downsampling.vhd	/home/carlos /devel /edelweiss /src/rx /tap.vhd	/home/ca /devel /edelwei /src/tx /bit2symb
Req4	-	-	-	-	-	-	-	-
Req4.1	-	-	-	-	-	-	-	-
Req4.2	-	-	-	-	-	-	-	-
Req4.3	-	-	X	-	-	-	-	-
Req1	-	-	-	-	-	-	-	-
Req2	-	-	-	-	-	-	-	X
Req3	-	-	-	-	-	-	-	-

Figura 5.8 Traceability Matrix Edelweiss.

5.3 Minitracer aplicado al propio Minitracer

El segundo tipo de prueba que se le pasó fue aplicar el propio trazador de requisitos a el mismo, de manera que pudiésemos saber si todos los requisitos que habían sido anotados en el fichero *csv* se habían cumplido y si además, algunos test que se generaron por defecto para que su salida siempre fuese correcta se leían correctamente del fichero que generaba *CTest.* De esta forma en la página de resumen del estado se puede observar que las 3 barras están al completo, habiendo cumplido por tanto los objetivos especificados. Además, en los marcadores superiores se pueden apreciar datos relevantes como ya se explicó antes y que corresponden a número de requisitos, tests, archivos csv y ficheros de código.

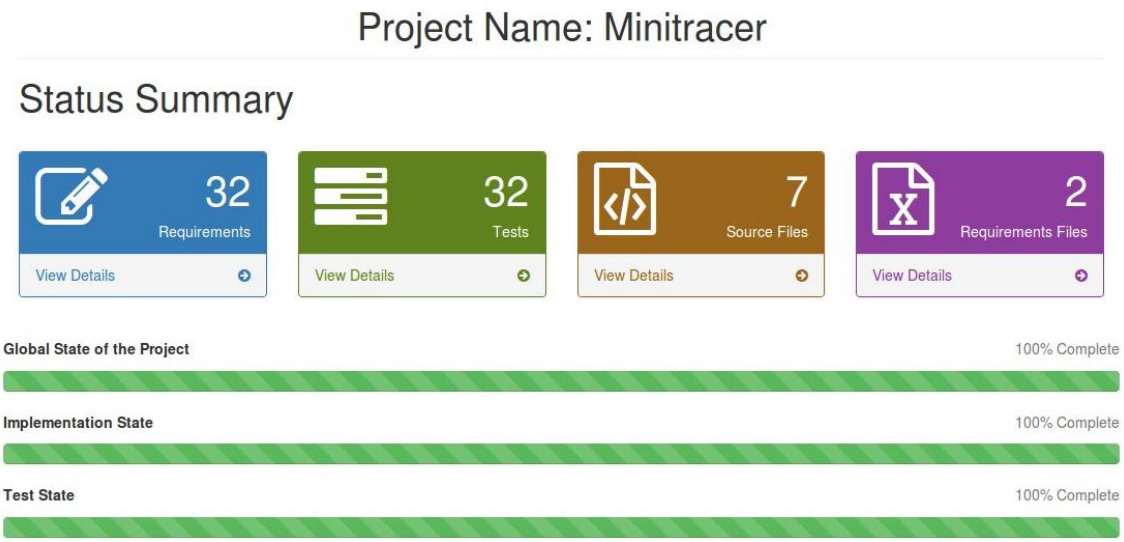
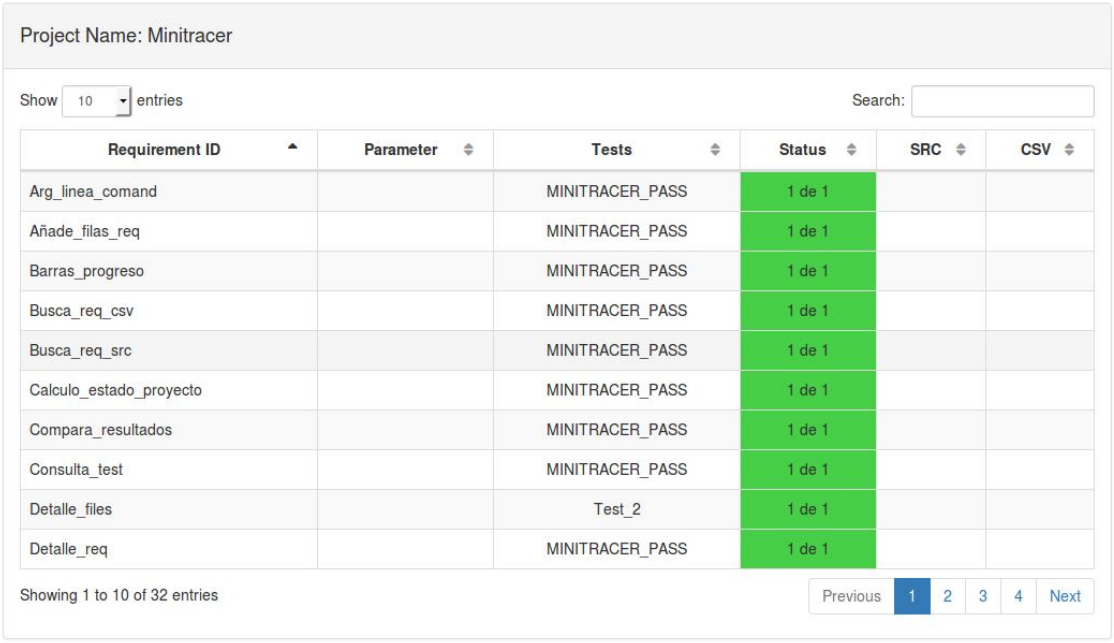


Figura 5.9 Status Summary minitracer.

Las imágenes que se ven a continuación son un recorrido por las diferentes pestañas del menú de la aplicación web para que se puedan apreciar los resultados obtenido en este caso. En la Figura 5.10 se puede ver con detalle el estado de cada requisito y los tests que pasan cada uno. Anotar que los tests denominados *<MINITRACER_PASS>* son tests que se han pasado manualmente a través de inspección visual y posteriormente se han marcado como correctos en el proyecto.


Project Name: Minitracer

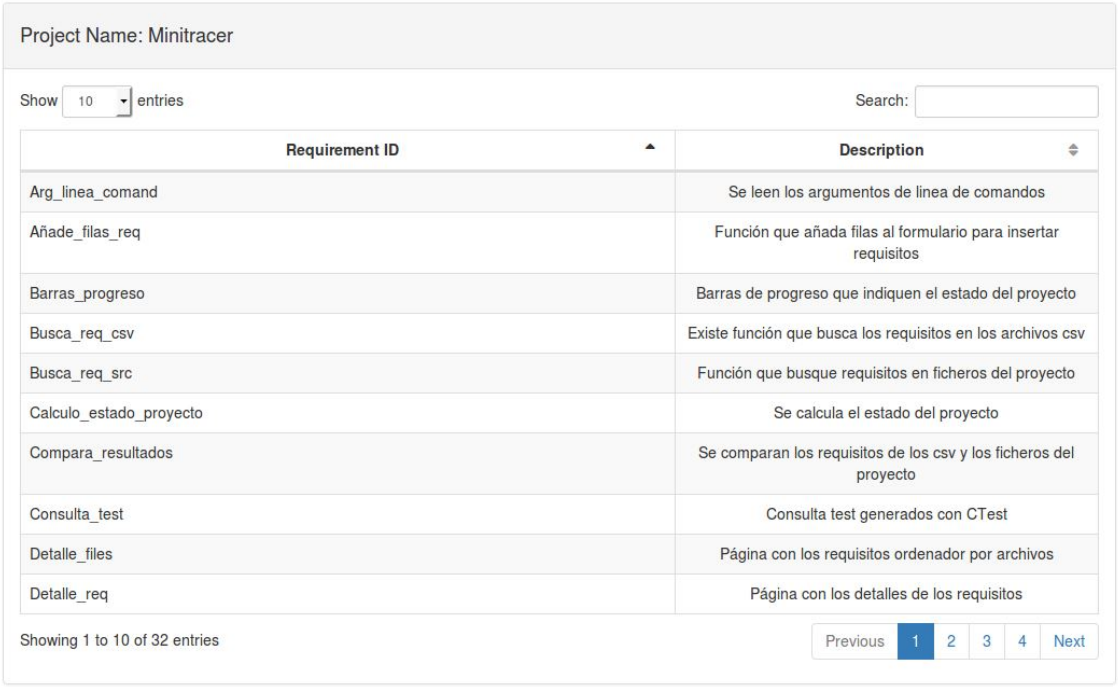
Show 10 entries Search:

Requirement ID	Parameter	Tests	Status	SRC	CSV
Arg_linea_comand		MINITRACER_PASS	1 de 1		
Añade_filas_req		MINITRACER_PASS	1 de 1		
Barras_progreso		MINITRACER_PASS	1 de 1		
Busca_req_csv		MINITRACER_PASS	1 de 1		
Busca_req_src		MINITRACER_PASS	1 de 1		
Calculo_estado_proyecto		MINITRACER_PASS	1 de 1		
Compara_resultados		MINITRACER_PASS	1 de 1		
Consulta_test		MINITRACER_PASS	1 de 1		
Detalle_files		Test_2	1 de 1		
Detalle_req		MINITRACER_PASS	1 de 1		

Showing 1 to 10 of 32 entries Previous 1 2 3 4 Next

Figura 5.10 Requirements Status minitracer.

En la Figura 5.11 se muestra una descripción de cada test.


Project Name: Minitracer

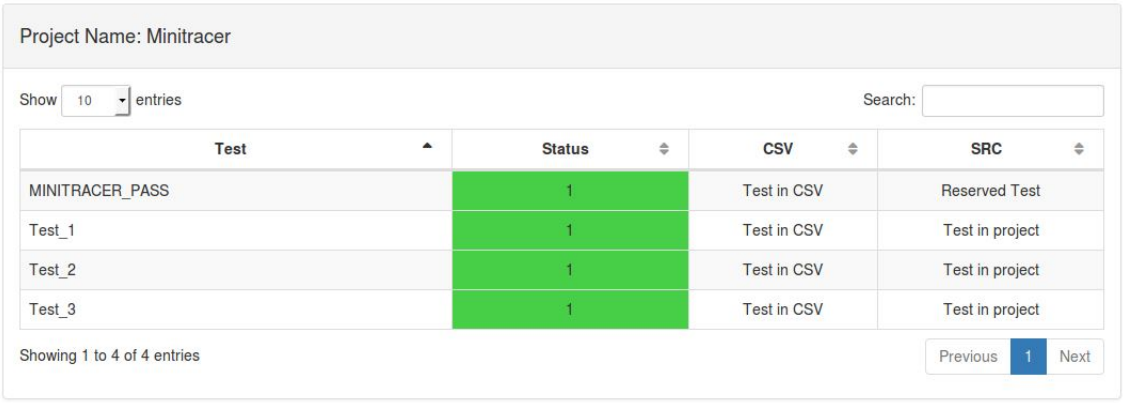
Show 10 entries Search:

Requirement ID	Description
Arg_linea_comand	Se leen los argumentos de linea de comandos
Añade_filas_req	Función que añada filas al formulario para insertar requisitos
Barras_progreso	Barras de progreso que indiquen el estado del proyecto
Busca_req_csv	Existe función que busca los requisitos en los archivos csv
Busca_req_src	Función que busque requisitos en ficheros del proyecto
Calculo_estado_proyecto	Se calcula el estado del proyecto
Compara_resultados	Se comparan los requisitos de los csv y los ficheros del proyecto
Consulta_test	Consulta test generados con CTest
Detalle_files	Página con los requisitos ordenador por archivos
Detalle_req	Página con los detalles de los requisitos

Showing 1 to 10 of 32 entries Previous 1 2 3 4 Next

Figura 5.11 Requirements Description Minitracer.

En la Figura 5.12 se muestra una descripción del estado de los tests.


Project Name: Minitracer

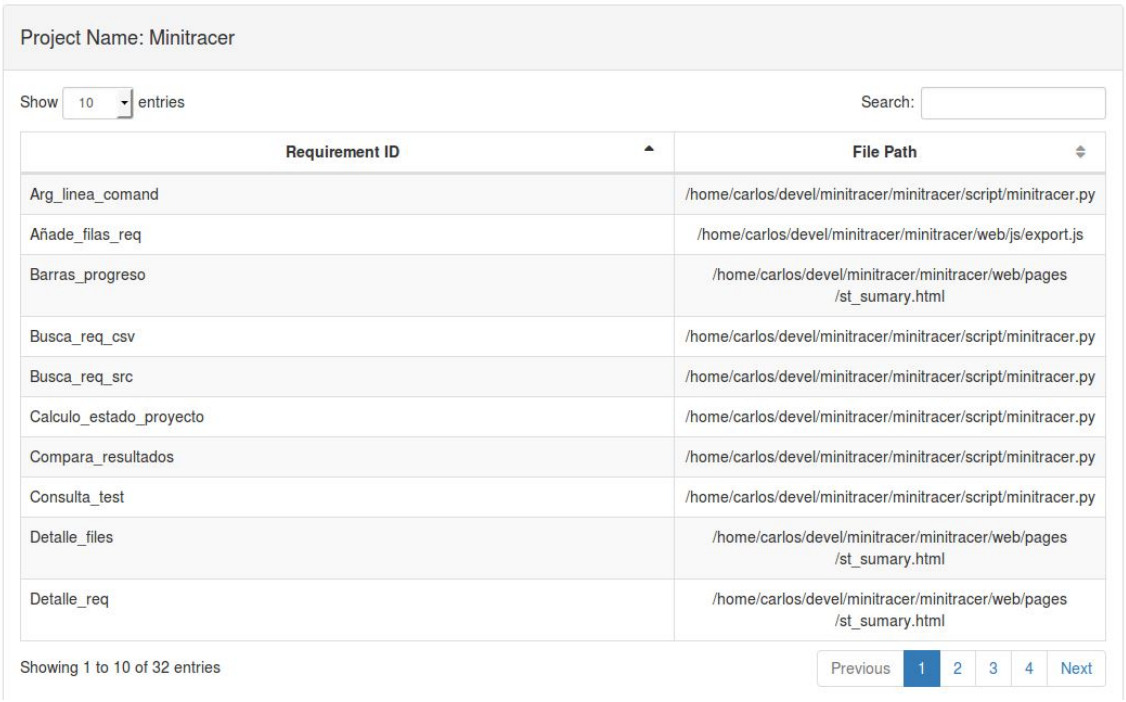
Show 10 entries Search:

Test	Status	CSV	SRC
MINITRACER_PASS	1	Test in CSV	Reserved Test
Test_1	1	Test in CSV	Test in project
Test_2	1	Test in CSV	Test in project
Test_3	1	Test in CSV	Test in project

Showing 1 to 4 of 4 entries Previous 1 Next

Figura 5.12 Tests Status minitracer.

En la Figura 5.13 se muestran los ficheros de código en los que se encuentra cada requisito.


Project Name: Minitracer

Show 10 entries Search:

Requirement ID	File Path
Arg_linea_comand	/home/carlos/devel/minitracer/minitracer/script/minitracer.py
Añade_filas_req	/home/carlos/devel/minitracer/minitracer/web/js/export.js
Barras_progreso	/home/carlos/devel/minitracer/minitracer/web/pages /st_sumary.html
Busca_req_csv	/home/carlos/devel/minitracer/minitracer/script/minitracer.py
Busca_req_src	/home/carlos/devel/minitracer/minitracer/script/minitracer.py
Calculo_estado_proyecto	/home/carlos/devel/minitracer/minitracer/script/minitracer.py
Compara_resultados	/home/carlos/devel/minitracer/minitracer/script/minitracer.py
Consulta_test	/home/carlos/devel/minitracer/minitracer/script/minitracer.py
Detalle_files	/home/carlos/devel/minitracer/minitracer/web/pages /st_sumary.html
Detalle_req	/home/carlos/devel/minitracer/minitracer/web/pages /st_sumary.html

Showing 1 to 10 of 32 entries Previous 1 2 3 4 Next

Figura 5.13 Tracing Files minitracer.

La Figura 5.14 muestra en su caso los ficheros **csv** en los que se encuentra cada requisito.

Project Name: Minitracer

Show 10 entries Search:

Requirement ID	CSV File Path
Arg_linea_comand	/home/carlos/devel/minitracer/minitracer/entradas/req_tracer /requisitos_py.csv
Añade_filas_req	/home/carlos/devel/minitracer/minitracer/entradas/req_tracer /requisitos_web.csv
Barras_progreso	/home/carlos/devel/minitracer/minitracer/entradas/req_tracer /requisitos_web.csv
Busca_req_csv	/home/carlos/devel/minitracer/minitracer/entradas/req_tracer /requisitos_py.csv
Busca_req_src	/home/carlos/devel/minitracer/minitracer/entradas/req_tracer /requisitos_py.csv
Calculo_estado_proyecto	/home/carlos/devel/minitracer/minitracer/entradas/req_tracer /requisitos_py.csv
Compara_resultados	/home/carlos/devel/minitracer/minitracer/entradas/req_tracer /requisitos_py.csv
Consulta_test	/home/carlos/devel/minitracer/minitracer/entradas/req_tracer /requisitos_py.csv

Figura 5.14 Requirements Files Minitracer.

En la Figura 5.15 se muestra la matriz de trazabilidad que enfrenta requisitos y tests. Como se puede observar en esta matriz, los requisitos que están relacionados con los test aparecen marcados con una “X” mientras que el resto de elementos que no tienen relación aparecen con un “-”.

###	MINITRACER_PASS	Test_3	Test_2	Test_1
Exporta_csv	X	-	-	-
Genera_matrices_web	-	X	-	-
Menu_web	X	-	-	-
Detalle_test	-	-	-	X
Fichero_path_src	X	-	-	-
Seleccion_addReq_genReport	X	-	-	-
Consulta_test	X	-	-	-
Añade_filas_req	X	-	-	-
Genera_datos_web	X	-	-	-
Detalle_req	X	-	-	-
Gestion_error_input	X	-	-	-
Gestion_test	X	-	-	-
Busca_req_csv	X	-	-	-
Escribe_resultado_test	X	-	-	-
Error_test_reservado	X	-	-	-
Arg_linea_comand	X	-	-	-

Figura 5.15 Verification Matrix minitracer.

Por último en la Figura 5.16 se muestra la matriz de trazabilidad que enfrenta requisitos y path. Como se puede observar en esta matriz, los requisitos que están relacionados con los archivos, es decir, los requisitos que están implementados dentro de un cierto archivo aparecen marcados con una “X”, mientras que el resto de elementos que no tienen relación aparecen con un “-”.

###	/home/carlos /devel /minitracer /minitracer /script /minitracer.py	/home/carlos /devel/minitracer /minitracer/script /lib_minitracer.py	/home/carlos /devel /minitracer /minitracer /web/pages /st_sumary.html	/home/carlos /devel /minitracer /minitracer /web/js /export.js	/home/carlos /devel /minitracer /minitracer /web/js /import.js	/home/carlos /devel /minitracer /minitracer /web/js /menu.js	/home/carlos /devel /minitracer /minitracer /web/js /parser /papaparse.js
Exporta_csv	-	-	-	X	-	-	-
Genera_matrices_web	-	-	X	-	-	-	-
Escribe_resultado_test	-	X	-	-	-	-	-
Detalle_test	-	-	X	-	-	-	-
Fichero_path_src	X	-	-	-	-	-	-
Seleccion_addReq_genReport	-	-	X	-	-	-	-
Consulta_test	X	-	-	-	-	-	-
Añade_filas_req	-	-	-	X	-	-	-
Genera_datos_web	-	-	-	-	X	-	-
Detalle_req	-	-	X	-	-	-	-
Gestion_error_input	X	-	-	-	-	-	-
Gestion_test	-	X	-	-	-	-	-
Busca_req_csv	X	-	-	-	-	-	-

Figura 5.16 Traceability Matrix Minitracer.

5.4 Situaciones particulares

En la siguiente sección se proponen algunas pruebas específicas al trazador de requisitos intentando abarcar cualquier situación que pudiese darse durante el desarrollo de un proyecto. La dinámica de representación de los resultados será la siguiente, en primer lugar se describirá la prueba realizada, y en segundo lugar se podrá observar el resultado en una figura.

5.4.1 Prueba 1

En la *Prueba 1* lo que se consideró fue la posibilidad de que hubiese requisitos en el fichero *csv* que no pudiesen ser encontrados en los archivos de código debido a que estos aún no estuviesen implementados. El resultado de esta prueba se puede ver en la Figura 5.17

Show 10 entries Search:

Requirement ID	Parameter	Tests	Status	SRC	CSV
Req_solo_csv1		MINITRACER_PASS	1 de 1	Req missing in project	
Req_solo_csv2		MINITRACER_PASS	1 de 1	Req missing in project	
Req_solo_csv3		MINITRACER_PASS	1 de 1	Req missing in project	
Req_solo_csv4		MINITRACER_PASS	1 de 1	Req missing in project	
Req_solo_csv5		MINITRACER_PASS	1 de 1	Req missing in project	
Req_solo_csv6		MINITRACER_PASS	1 de 1	Req missing in project	
Req_solo_csv7		MINITRACER_PASS	1 de 1	Req missing in project	
Req_solo_csv8		MINITRACER_PASS	1 de 1	Req missing in project	

Figura 5.17 Requisito no encontrado en proyecto.

5.4.2 Prueba 2

En la *Prueba 2* lo que se consideró fue la posibilidad de que hubiese requisitos en los archivos de código que no hubiesen sido añadidos al archivo *csv* o que no tuviesen exactamente la misma etiqueta. El resultado de esta prueba se puede ver en la Figura 5.18

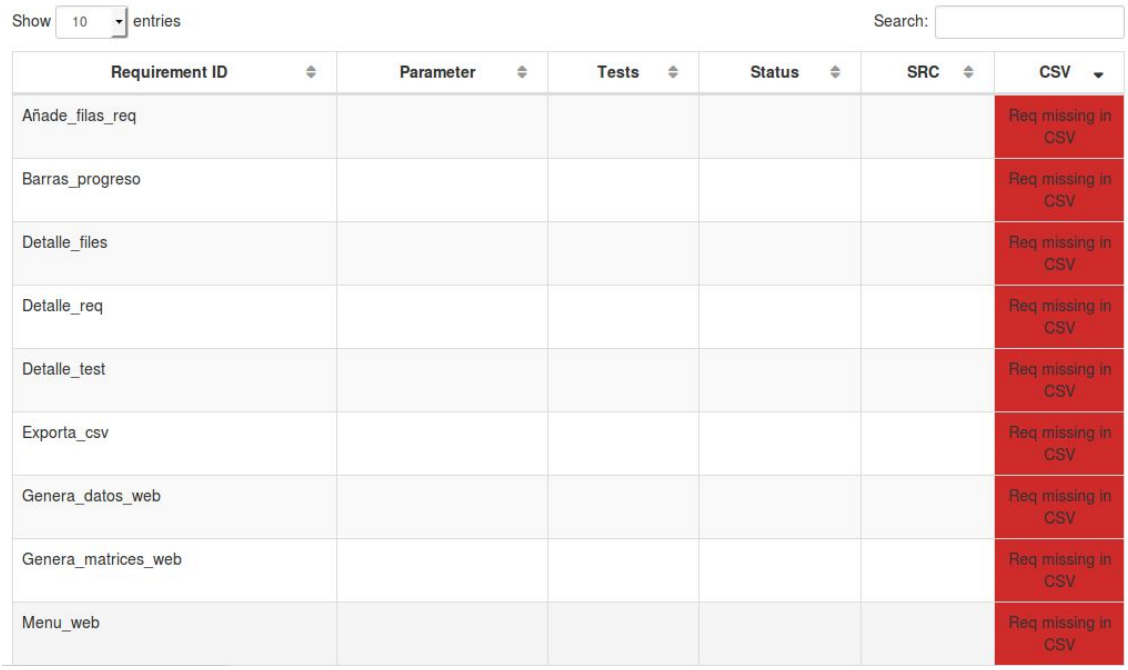


Figura 5.18 Requisito no encontrado en archivo csv.

5.4.3 Prueba 3

En la *Prueba 3* se consideró la posibilidad de que hubiese algún test en el archivo de requisitos que no hubiese sido implementado en el proyecto o que al ejecutar *CTest* estuviese siendo ignorado. El resultado de esta prueba se puede ver en la Figura 5.19

Figura 5.19 Test no encontrado en proyecto.

5.4.4 Prueba 4

En la *Prueba 4* lo que se quiso demostrar es que si existe algún test en el proyecto que no se ha tenido en cuenta en el fichero csv, éste lo indica. El resultado de esta prueba se puede ver en la Figura 5.20

Test_1	1	Test missing in CSV	Test in project
Test_2	1	Test missing in CSV	Test in project
Test_3	1	Test missing in CSV	Test in project

Figura 5.20 Test no encontrado en archivo csv.

5.4.5 Prueba 5

La *Prueba 5* realizada al proyecto fue referente al script, en ella se introducían los argumentos por línea de comando desordenados o incompletos. El resultado de esta prueba se puede ver en la Figura 5.21

```
carlos@carlos:~/devel/minitracer/py$
carlos@carlos:~/devel/minitracer/py$ ./minitracer.py ~/ruta_inventada ~/devel/edelweiss/requisitos.txt ~/inventado/edelweiss/build/

Error on input arguments
Input example:
        ./[PROGRAM_NAME] [FILE_WITH_REQ_PATHS] [FILE_WITH_SRC_PATHS] [PROJECT_PATH]

carlos@carlos:~/devel/minitracer/py$
```

Figura 5.21 Argumentos erróneos o incompletos por línea de comandos.

5.4.6 Prueba 6

En la *Prueba 6* lo que se quería saber es como reaccionaba nuevamente el script antes un fallo con los ficheros, en este caso fue que alguno de los ficheros, bien de requisitos o de código del proyecto, no existían o no tenían permisos de apertura y lectura. El resultado de esta prueba se puede ver en la Figura 5.22

```
carlos@carlos:~/devel/minitracer/py$
carlos@carlos:~/devel/minitracer/py$ ./minitracer.py ~/devel/edelweiss/requisitos.txt ~/devel/edelweiss/source.txt ~/devel/edelweiss/b
uild/
Error in open_src_files: File /home/carlos/devel/edelweiss/src/common/d_fff3.vhd not found
Write complete path
carlos@carlos:~/devel/minitracer/py$
```

Figura 5.22 Archivo no existente o sin permisos.

5.4.7 Prueba 7

La *Prueba 7* consistía en un caso en el que, por error, se intentasen pasar al trazador archivos de requisitos pertenecientes a proyectos distintos. En ese caso el trazador debería dar un error y detener la ejecución del script. El resultado de esta prueba se puede ver en la Figura 5.23

```
carlos@carlos:~/devel/minitracer/py$
carlos@carlos:~/devel/minitracer/py$ ./minitracer.py ~/devel/minitracer/build/requisitos.txt ~/devel/minitracer/build/source.txt /home
/carlos/devel/minitracer/build/
Error in "Project_Name" field in CSV files
carlos@carlos:~/devel/minitracer/py$
```

Figura 5.23 Requisitos de proyectos distintos.

5.4.8 Prueba 8

Por último, la *Prueba 8* consistía en un caso en el que, por error, el proyecto contaba con algún test que tenía como nombre alguno de los reservados para los test del trazador. El resultado de esta prueba se puede ver en la Figura 5.24

```
carlos@carlos:~/devel/minitracer/minitracer/script$ ./minitracer.py ~/devel/minitracer/minitracer/entradas/Casos_especiales/requisitos
.txt ~/devel/minitracer/minitracer/entradas/Casos_especiales/source.txt ~/devel/minitracer/build/
Error on "Test_Name" in your project. If your have tests with any of the next reserved names, please change them
        MINITRACER_PASS
        MINITRACER_FAIL
        MINITRACER_PASS
```

Figura 5.24 Test con nombre reservado en proyecto.

6 Conclusiones y Trabajos Futuros

Unas buenas herramientas y actitudes en la gestión de proyecto nos ahorrará tiempo, esfuerzo, trabajo... en resumen, dinero

ANÓNIMO

En este capítulo se describen algunos de los posibles trabajos futuros que se podrían hacer al proyecto realizado para mejorar su funcionamiento, añadir funcionalidades, cambiar los estilos de la interfaz web o adaptar el trazador de requisitos a diferentes normativas según convenga. Por otro lado también se presentan las conclusión del trabajo realizado.

6.1 Trabajos Futuros

En esta sección se enumeran una serie de trabajos futuros que se pueden realizar para mejorar el trazador de requisitos. Además de los que se van a mencionar, como la aplicación se ha realizado con la idea de que sea software libre, cualquier usuario que lo desee puede añadir las funcionalidades que desee oportunas siempre que sirvan para mejorar el citado software y sigan en la línea de que es un software para diseños HDL. En cualquier caso deberán reportar dichas modificaciones al autor de dicho proyecto para que sean tenidas en cuenta y añadidas a la versión oficial del programa. Los trabajos futuros que se han pensado son los siguientes:

1. Adaptación del software para cumplir con la normativa *DO-254* para proyectos aeroespaciales.
2. Creación de una hoja de estilos para las matrices de trazabilidad para que los títulos se las columnas se lean en vertical y con esto se pueda disminuir el tamaño horizontal de la matriz.
3. Creación de una función de *JavaScript* para poder importar requisitos al formulario de la aplicación web y poder añadir nuevos requisitos a los ya existentes.
4. Creación de un logo específico para *minitracer.*
5. Añadir la opción de que no sea necesario tener la carpeta *build* en el proyecto para la correcta ejecución del software implementado.
6. Posibilidad de que el software funcione correctamente sin necesidad de ejecutar con anterioridad *CMake* y *CTest.*
7. Automatizar los test propios de Minitracer de forma que no haya que realizar la verificación de la funcionalidad del mismo de forma manual como se ha hecho hasta ahora. Esto será especialmente útil en el caso de realizar mejoras o ampliaciones del programa.

8. Algoritmo para procesar el formulario de entrada de requisitos para alertar al usuario si hay campos obligatorios que deben contener algún valor.

6.2 Conclusiones

Una vez realizado el proyecto del trazador de requisitos para diseños HDL y después de todo lo aprendido durante su desarrollo en el ámbito tanto de la *gestión de requisitos,* la *gestión de proyectos* y la *ingeniería de software* así como la destreza adquirida en los diferentes lenguajes de programación empleados se ha llegado a una serie de conclusiones.

Por un lado, se ha visto la importancia y los beneficios que tiene realizar una buena gestión de los proyectos ya que ayuda tanto a los desarrolladores como al resto de personas implicadas directa o indirectamente en el proyecto a saber cual es el estado del mismo en cada momento de su etapa, qué se encuentra realizado, que falta por realizar, de qué tiempo se dispone, la parte del presupuesto que se lleva consumida, etc. Es también la mejor forma de poder gestionar los cambios producidos durante la fase de desarrollo y poder hacer frente a los imprevistos que surjan tanto en esta fase como en etapas posteriores. Dentro de esta gestión del proyecto, se ha visto la gran utilidad que tiene la gestión de los requisitos ya que nos permite obtener varias herramientas para conocer también el estado de nuestro proyecto en base a lo que el cliente pidió a modo de requisitos. Es también una forma de poder generar documentos que expliquen de manera resumida, precisa y fácil de entender en qué fase de desarrollo se encuentra el trabajo sin necesidad de que la persona que lea dicho documento tenga que estar directamente involucrada en el desarrollo. A parte de la parte del conocimiento del estado de desarrollo, la realización de una buena gestión de los requisitos y del proyecto en general tiene otros beneficios como pueden ser los económicos ya que ante cualquier imprevisto se puede reaccionar de manera más rápida y con mayores expectativas de éxito que si no se ha realizado una buena gestión del proyecto y de los requisitos a través de un trazador de los mismos. En gran parte, estos beneficios económicos que se pueden conseguir dictan en la mayoría de los casos si un proyecto merece la pena o no ya que, por muy bueno que sea un proyecto, si se ha llegado a invertir más incluso de lo que después se espera recuperar habrá sido un completo fracaso. También otro beneficio que se puede obtener y que está íntimamente ligado con el económico es el tiempo, con una buena organización se puede disminuir el tiempo de resolución de un problema a mucho menos de la mitad y como bien se sabe, en la mayoría de los casos, tiempo es igual a dinero.

Por otro lado, durante la realización de este proyecto se han aprendido un gran número de cosas, no directamente ligadas a los beneficios de tener un trazador de requisitos para los proyectos sino más bien referentes a las buenas prácticas en el desarrollo de proyectos. Como ya en la asignatura de *Proyectos de Sistemas Electrónicos* de 4º curso del *Grado en Ingeniería de las Tecnologías de Telecomunicaciones* nos introdujo el profesor D. Hipólito Guzmán Miranda, el uso de un repositorio como puede ser *Git,* o de herramientas como pueden ser *Mantis* para *bugtracking* o *Doxygen* para la generación de documentación facilitan mucho la realización de un proyecto y algunas como pueden ser el caso de los repositorios, entre otras cosas, previenen de sobre-escrituras de versiones finales o permiten recuperar versiones del proyecto anteriores o incluso recuperar ficheros borrados o, lo que sería más general, proteger todos los ficheros de nuestro proyecto de nosotros mismos, pueden salvar en muchos casos que un proyecto esté terminado en fecha o tenga retrasos inesperados. Junto a estas herramientas, también se explicó una serie de buenas prácticas en el desarrollo de proyectos, algunas de las cuales se han tenido en cuenta durante la realización de este trazador de requisitos y se ha podido comprobar de primera mano que además de ahorrar tiempo, te ahorran de más de un quebradero de cabeza. Algunas de estas buenas prácticas han sido la generación de backups semi-automáticos, a diferencia de lo que se explicó como backups automáticos, en este caso se realizaban a mano ya que se realizaban copias a mano en un medio extraíble todas las noches, después de hacer push de los cambios generados durante el día se copiaba esta última versión en un medio extraíble. También, se pudo observar que utilizando la buena práctica de hacer una planificación realista se ahorra el estrés de los últimos días.

Junto a estas conclusiones a nivel del uso de un trazador de requisitos y del empleo de herramientas y de buenas prácticas en el desarrollo de proyectos se han sacado una serie de conclusiones también a raíz de los lenguajes de programación empleados. En primer lugar, se ha descubierto el lenguaje *Python,* un lenguaje que se conocía solo de oídas al principio de este desarrollo y que cuando se empezó a aprender y usar se pudo

ver que es un lenguaje muy potente con infinidad de recursos para hacer casi cualquier tipo de aplicación software que se necesite de una manera sencilla y a muy alto nivel ya que tiene funciones definidas para casi todo. En segundo lugar, y referente a las hojas de estilo *CSS,* se ha podido observar la cantidad de estilos que se pueden aplicar a una página *HTML* y como cambia el aspecto de la misma cuando se sabe crear una hoja de estilos avanzada y cuando se sabe lo básico. Por último y referente al lenguaje *JavaScript,* se han aprendido varias utilidades nuevas a la vez que se han descubierto funciones que permiten parsear archivos *csv,* desde los propios en una máquina local hasta los que se encuentran en diferentes servidores, a la vez que se ha visto la potencia que puede llegar a tener dicho lenguaje cuando se adquiere un conocimiento medio-avanzado.

Por todo esto considero que, a pesar de que en un principio el proyecto del trazador de requisitos se convirtió en algo bastante complicado y en momentos casi inalcanzable debido a la falta de conocimientos, una vez se fueron superando estos impedimentos se ha aprendido, no solo lenguajes de programación nuevos y herramientas nuevas sino también los beneficios que tiene poseer una serie de herramientas que, aunque no influyan directamente en el resultado explícito de un proyecto, son elementos dictaminadores a la hora de poder sacar un proyecto adelante y poder resolver cualquier imprevisto que surja en el camino, pero, no solo queda ahí lo aprendido, una de lo que podría ser de las partes más importantes ha sido el poder experimentar como, con el uso de sólo algunas buenas prácticas en el desarrollo de proyectos, el trabajo realizado se fue convirtiendo cada vez en algo más sencillo de controlar y estimulante en su desarrollo.

Índice de Figuras

Índice de Códigos

Bibliografía

[1] [Online]. Available: https://en.wikipedia.org/wiki/Test-driven_developmentx

[2] [Online]. Available: https://en.wikipedia.org/wiki/Requirements_management

[3] [Online]. Available: https://en.wikipedia.org/wiki/Requirements_traceability

[4] [Online]. Available: https://en.wikipedia.org/wiki/Traceability_matrix

[5] [Online]. Available: https://en.wikipedia.org/wiki/Functional_requirement

[6] [Online]. Available: https://en.wikipedia.org/wiki/Non-functional_requirement

[7] H. G. Miranda, *Tema 4 Extracción de Requisitos*, 2016.

[8] [Online]. Available: https://cmake.org/Wiki/CMake/Testing_With_CTest#Introduction

[9] [Online]. Available: https://msdn.microsoft.com/en-us/library/dd420548.aspx

[10] G. Eduardo, U. G. German, and G. P. Liliana, *REASEM: Requirement management tool*, 2 de septiembre de 2009.

[11] [Online]. Available: http://www.ipcorp.com.ar/blog/2008/12/11/osrmt-open-source-requirements-management-tool/

[12] [Online]. Available: http://www-03.ibm.com/software/products/es/ratidoor

[13] [Online]. Available: https://www.aldec.com/en

[14] [Online]. Available: https://www.polarion.com/

[15] [Online]. Available: http://www8.hp.com/es/es/software-solutions/requirements-management/

[16] [Online]. Available: http://www.visuresolutions.es/

[17] G. Eduardo, U. G. German, and G. P. Liliana, "Reasem: Requirement management tool," 2 de septiembre de 2009.

[18] O. Gotel and A. Finklestein. (1994) An analysis of the requirements traceability problem.

[19] [Online]. Available: https://www.techopedia.com/definition/27291/many-to-many-relationship

[20] In science, computing, and engineering, a black box is a device, system or object which can be viewed in terms of its inputs and outputs (or transfer characteristics), without any knowledge of its internal workings. [Online]. Available: https://en.wikipedia.org/wiki/Black_box

[21] In software testing, a test harness or automated test framework is a collection of software and test data configured to test a program unit by running it under varying conditions and monitoring its behavior

and outputs. [Online]. Available: https://en.wikipedia.org/wiki/Test_harness

[22] [Online]. Available: https://cmake.org/Wiki/CMake/Testing_With_CTest#Simple_Testing

[23] [Online]. Available: https://cmake.org/Wiki/CMake_Scripting_Of_CTest

[24] [Online]. Available: https://www.lsi.us.es/descargas/descarga_programas.php?id=3

[25] "Heler: Una herramienta para la ingeniería de requisitos automatizada," el proceso Unificado es una metodología para el proceso de desarrollo de software; contiene un conjunto de actividades que transforman los requisitos de usuario en un sistema software. Es un proceso dirigido por casos de usos, centrado en la arquitectura, iterativo e incremental. [Online]. Available: http://biblat.unam.mx/es/revista/entramado/articulo/heler-una-herramienta-para-la-ingenieria-de-requisitos-automatizada

[26] Datasheet. [Online]. Available: https://www.aldec.com/en/products/requirements_management/spectracer

[27] Datasheet. [Online]. Available: https://www.polarion.com/hubfs/Docs/Fact-sheets/Polarion_REQUIREMENTS_Fact-Sheet.pdf?__hssc=&__hstc=90891525.28276f7f8c4b700c16d968e5edae32a6.1464858997407.1464858997407.1464869396493.2&__hsfp=2354658137&hsCtaTracking=3bdfd088-dd11-46f5-94cd-62f5768435e7|5a83f51c-23b4-4fb9-8d25-8ee07cb4a3df

[28] Datasheet. [Online]. Available: http://www.borland.com/Borland/media/Resources/Data%20sheets/BDS-Atlas-3.pdf?ext=.pdf

[29] Datasheet. [Online]. Available: http://www.borland.com/Borland/media/Resources/Data%20sheets/BDS-Caliber_11-4.pdf?ext=.pdf

[30] *HPE Quality Center Software" Quality management software delivers highest quality applications.* [Online]. Available: http://www8.hp.com/es/es/software-solutions/quality-center-quality-management/index.html

[31] *Accelerate your business: HPE Application Lifecycle Management Software.* [Online]. Available: http://www8.hp.com/es/es/software-solutions/alm-software-development-testing/index.html

[32] La lista ha sido obtenida de la sección "hoja de producto. [Online]. Available: http://www.visuresolutions.es/herramienta-ingeniera-requisitos

[33] J. B. Rojas, *Implementación y Análisis de SEU de un transmisor de comunicaciones inalámbricas para aplicaciones intra-satélite*, 2015.